PROGRAMAS PRACTICOS PARA EL SPECTRUM

UNA BIBLIOTECA DE MODULOS Y SUBRUTINAS

D. LAWRENCE



PROGRAMAS PRACTICOS PARA EL SPECTRUM

Editorial Gustavo Gili, S. A.

08029 Barcelona Rosellón, 87-89. Tel. 322 81 61
28006 Madrid Alcántara, 21. Tel. 401 17 02
1064 Buenos Aires Cochabamba, 154-158. Tel. 361 99 98
México, Naucalpan 53050 Valle de Bravo, 21 - Tels. 560 60 11 y 13
Bogotá Diagonal 45 N.° 16 B-11. Tel. 245 67 60
Santiago de Chile Vicuña Mackenna, 462, Tel. 222 45 67

PROGRAMAS PRACTICOS PARA EL SPECTRUM

UNA BIBLIOTECA DE MODULOS Y SUBRUTINAS

D. LAWRENCE

GG

Título original
The Working Spectrum
A Library of practical subroutines and programs

Publicado originalmente en inglés en 1982 por Sunshine Books (An imprint of Scot Press Ltd.) 12-13 Little Newport Street - London WC2R 3LD

Versión castellana de Jordi Abadal Berini, Ingeniero Industrial (texto) y Antoni Llaverías Santacana, Ingeniero Industrial (programas y listados)

Printed in Spain
ISBN: 84-252-1281-2
Depósito legal: B. 19.472-1986
Fotocomposición: Tecfa, S. A. - Barcelona
Impresión: Hurope, S. A. - Recaredo, 2 - Barcelona

Indice

Nota del traductor

Los programas de este libro se han listado mediante una impresora Riteman F+ en un Spectrum con Interface 1. La ROM del Spectrum fue reemplazada por una EPROM adaptada al hardware del mismo, que contenía una modificación para hacer posible la ejecución de los comandos COPY, LLIST y LPRINT tal como se efectuarían con el ZX Printer, conservando además la relación 1:1 horizontal/vertical con la que aparecen en pantalla.

Hay un punto a tener en cuenta con los listados del programa que utilizan UDG:

1) Ejecutar primero la rutina correspondiente del programa para que al pulsar las teclas en modo gráfico aparezcan en pantalla tal como en el listado del libro. Es necesario hacer esto sólo en la fase de entrar el programa.

El contenido en detalle

Esta colección de programas se ha agrupado en seis capítulos:

1. *Búsqueda de un archivo. El Spectrum como archivador*

1.1 Archivo: Este es un programa flexible que permite manejar cualquier archivo que contenga registros con una estructura regular de elementos tales como nombre, dirección, número de teléfono y edad. Se puede buscar, guardar, obtener, corregir y borrar cualquier número regular de elementos.

2. *Realización del presupuesto. El Spectrum como banquero*

En el último capítulo hemos estudiado las técnicas del manejo de datos no numéricos. En este capítulo trataremos con números y en particular con dinero.

2.1 Presupuesto: Este es un programa de presupuestos muy flexible y bien formateado.

2.2 Contabilidad: Este sencillo programa le ayudará a visualizar el estado de sus cuentas de una forma clara y fácil.

2.3 Cuentas bancarias: Este programa es una inteligente herramienta que le permitirá mantener su propio registro financiero en forma parecida a la utilizada en los estados de cuentas bancarios. También utilizaremos por vez primera líneas multisentencia en el programa.

3. *Dibujar. Gráficos del Spectrum*

3.1 Caracteres: Algunos de ustedes quizá ya hayan escrito un programa para definir caracteres. Sin embargo, éste es un buen programa de utilidad para aquellos que no lo hayan hecho y constituye una sencilla introducción a algunas de las técnicas que utilizaremos en programas posteriores.

3.2 Diccionario: Este corto programa aumenta en gran manera la utilidad del generador de caracteres, permitiéndole crear un diccionario de nuevos caracteres.

3.3 Tangram: Este antiguo juego chino le dará una idea de las figuras geométricas que pueden dibujarse sin necesidad de funciones matemáticas complejas.

3.4 Artista: Si va a realizar programas que utilicen dibujos como

parte de las visualizaciones en pantalla entonces, con la ayuda de este programa, podrá crear dibujos de una forma sencilla y después guardarlos.

3.5 Diseño: Este programa le permitirá definir un dibujo de hasta 65536 × 65536 pixels, añadir y borrar, examinarlo a varias escalas y girarlo, en su totalidad o parte del mismo, sobre la pantalla.

4. *Educación fácil. El Spectrum como tutor doméstico*

4.1 Respuesta múltiple: Se trata de un programa de elección múltiple que puede guardar hasta mil preguntas y respuestas distintas. El programa guarda sus datos en una serie de tablas bidimensionales y muestra muchas técnicas nuevas para el manejo de datos. Es un programa sorprendentemente apasionante.

4.2 Palabras: Este programa es una variación del de respuesta múltiple, cuya única diferencia real es que las preguntas se realizan mediante dibujos. Se puede utilizar para aprender a leer.

4.3 Geografía: Es un programa poco complicado que comprueba de una forma efectiva sus conocimientos de geografía. El programa se construye mediante módulos que provienen del programa Artista, del capítulo 3.

5. *Programas de utilidad. Una colección de rutinas variadas*

5.1 Calculadora: Se trata de un programa que le permitirá entrar, de una forma fácil, una gran variedad de fórmulas y variables y utilizarlas sin realizar cálculos repetitivos. Los resultados se visualizan en tablas fáciles de interpretar.

5.2 Calorías: Si desea contar las calorías de sus comidas encontrará de utilidad este programa. Sin embargo, su principal objetivo es mostrar lo rápido que puede ser el construir y utilizar diccionarios de elementos y de cantidades asociadas.

5.3 Gráficas: Este corto programa es un trazador de gráficas de propósito general que le permite definir las unidades de ambos ejes, tanto respecto al nombre como a su longitud y entrar datos ordenados o desordenados para crear una gráfica lineal.

5.4 Renumeración: Un sofisticado programa de renumeración que probablemente utilizará más que cualquier otro. Renumera también los GOTO y los GOSUB.

5.5 Archivo 2: Es una modificación más avanzada del programa Archivo original, del capítulo 1. Puede utilizarse como programa de base de datos en el caso de que la estructura de los datos que haya que manejar sea predecible en cuanto a su estructura o longitud.

5.6 Mecanografía: No todos los programas tienen que tener cientos de líneas. Este le ayudará a aprender y a practicar la mecanografía.

6. *Y finalmente un poco de diversión. Juegos para el Spectrum*

6.1 Misil: Un juego que requiere cálculos reales.

6.2 Cacería: Un emocionante juego en el que hay que cazar una presa invisible.

6.3 Clasificación de palabras: Es una sencilla rutina de clasificación de cadenas alfanuméricas que puede utilizarse en los juegos de palabras. Puede adaptarse fácilmente para que realice clasificaciones numéricas.

Introducción

Este libro fue realizado en un intento de llenar un gran hueco. El hueco era la ausencia de trabajos destinados a cumplir el sueño del nuevo propietario de un microordenador que consiste en que su máquina no sea únicamente un juguete, ni tampoco una introducción educacional a la «edad del silicio», sino una herramienta potente, capaz de realizar todo tipo de tareas y que abra todo tipo de posibilidades. La mayoría de los libros contienen programas demasiado triviales o presuponen un gran deseo —o quizás una gran capacidad— para experimentar.

Yo he querido escribir un libro basado en una sólida colección de programas en áreas tales como el almacenaje de datos, las finanzas, gráficos, cálculos y aplicaciones domésticas y educativas. Las discusiones sobre las técnicas de programación irán apareciendo a partir de los propios programas de forma parecida, quizás, a un artículo de revista asociado al programa. Espero que encuentre que el libro que ha salido de este deseo sea de utilidad, no sólo como una forma de aprender nuevas técnicas de programación, sino también como una colección de programas, que ofrecen un amplio rango de aplicaciones que de otra forma estarían abiertas únicamente a aquéllos dispuestos a comprar programas comerciales muy caros o capaces de escribir programas sustanciales por sí mismos mediante su experiencia en programación.

Este conjunto de programas pueden clasificarse, más o menos, en cinco grupos que se dan a continuación. He tenido en cuenta el peligro que podría representar el presentar simplemente una masa indigerible de programas. Por este motivo, además del hecho de ser muy adecuado para el Spectrum, todos los programas se han escrito en forma modular. En el texto se dan instrucciones completas para la comprobación de cada módulo antes de pasar al siguiente.

Una sección típica del libro tiene la siguiente forma:

Título del programa: por ejemplo, el Spectrum como archivador

A continuación sigue una introducción general de la sección, estableciendo las ventajas y desventajas de utilizar el Spectrum para este tipo de aplicación. Esto va seguido por una o más secciones que se refieren a las técnicas de programación específicas utilizadas para

esta aplicación. En éstas se pueden incluir técnicas de búsqueda, tablas de punteros y el guardar datos en variables alfanuméricas con más de una dimensión. Después se examina con detalle cada uno de los módulos utilizados en el programa.

El texto que figura a continuación de los módulos se divide en tres secciones:

a) Un breve análisis del propósito del módulo y de cualquier punto de interés particular que se haya planteado.
b) Un comentario sobre las propias líneas, identificando los ejemplos de las distintas técnicas y señalando la unidad en la que el módulo puede incluirse de una forma natural.
c) Se dan sugerencias para realizar algunas comprobaciones sencillas por si quiere comprobar por sí mismo que el módulo no es muy incorrecto antes de entrar el siguiente.

Realice estas comprobaciones de una forma seria; pueden ahorrarle muchos dolores de cabeza posteriormente.

Una vez completa la lista de módulos hay un breve resumen de las principales lecciones que puedan haberse aprendido durante el proceso de entrada del programa.

Una de las ventajas de este planteo es que los módulos podrán utilizarse en otros contextos una vez comprendida su función, una posibilidad que todavía es más atractiva por la capacidad que tiene el Spectrum de mezclar programas.

La posibilidad que tiene el Spectrum para aceptar varias sentencias en una única línea de BASIC puede ser una ventaja o todo lo contrario. Para que fuese más fácil su utilización y edición, he evitado utilizar líneas multisentencia cuando se trataba de nuevas aplicaciones.

Por lo tanto existen tres formas de utilizar este libro:

1) Como una forma nueva de aprender a programar.
2) Como un paquete de útiles programas de aplicación.
3) Como una gran colección de módulos, o subrutinas que pueden utilizarse en un número ilimitado de otros programas. Al final del libro encontrará un índice extensivo que cubre las funciones de cada uno de los módulos de programación.

En otras palabras, puede considerar este libro como una introducción a la programación más sofisticada, como una biblioteca de programas de utilidad o como un manual de referencia de programación. Sin embargo, para obtener el máximo de este libro, le recomiendo que empiece por el principio y vaya abriéndose el camino a través de él.

Como técnica de aprendizaje le será de utilidad el leer el resumen del libro un par de veces, para tener una idea del libro completo. Cuando empiece cada sección, lea la introducción, sáltese lo siguiente y lea el resumen. Después lea la sección completa. De esta forma recordará mucho más el contenido.

1. Búsqueda de un archivo. El Spectrum como archivador

1.1 Archivo

Tarde o temprano, la mayoría de los poseedores de un micro se dan cuenta de que su nuevo amigo digital está realmente a sus anchas cuando almacena información, la procesa y la presenta en formas que serían extremadamente laboriosas si fueran manuales. Entonces empiezan a escribir sencillos programas que guarden los nombres y direcciones de sus amigos, o cataloguen su álbum de sellos. Terminan con media docena de programas, cada uno dedicado a una utilización, pero funcionando todos con el mismo método.

En este capítulo inicial daremos un gran salto y examinaremos cómo puede escribirse un sencillo programa que satisfaga una gran variedad de tareas de archivo distintas, sin la necesidad de estar continuamente reescribiendo un programa cada vez que aparece una nueva aplicación.

Antes de escribir el programa tendremos que decidirnos por una forma económica de guardar los datos que queremos archivar. Los programas pequeños pueden saltarse este problema, ya que es muy poco probable que utilicen la totalidad de la memoria disponible. Estos programas pequeños, por ejemplo, pueden declarar una tabla cuyas dimensiones sean 5Ø, 4, 2Ø. Esto nos permitirá guardar hasta 5Ø registros, cada uno constituido por cuatro elementos, y cada elemento con un máximo de 2Ø caracteres de largo. La ventaja de esto es que cada registro del archivo sería claramente identificable, ya que el registro X estaría constituido, en el caso de que la tabla se llamase A$, por A$(X,1), A$(X,2), A$(X,3) y A$(X,4).

La desventaja de este método es que, para la mayoría de las aplicaciones de archivo, es muy probable que dé como resultado un gasto enorme de espacio de memoria de la cantidad limitada de memoria disponible. La razón de esto es sencillamente que la longitud del espacio fijo que se asigna a cada elemento debe ser adecuado para el elemento más largo que pueda entrarse. Por ejemplo, si se quieren guardar los nombres de los amigos y uno de ellos tiene el apellido Fernández de Montoya, entonces tendrá que reservar al menos 21 caracteres para cada apellido, independientemente del hecho de que el resto de sus amigos no tengan ni mucho menos un apellido

tan impresionante. El espacio asignado a la mayoría de los nombres estaría por lo tanto medio vacío.

Este es un problema que aparece en cualquier método de almacenaje que asigne una cantidad fija de memoria a cada elemento, independientemente de su tamaño. Pero si no se asigna a cada registro una cantidad de memoria fija, entonces el archivo en que están guardados los datos no quedará dividido de una forma regular. Esto hace más difícil para el programa el seguimiento de la posición de los registros individuales o incluso el identificar dónde termina un registro y dónde empieza otro.

Tomemos el siguiente ejemplo de dos registros en un archivo, a los que se ha asignado exactamente la cantidad correcta de espacio:

```
PEREZJUAN331255645677C.MAYORLOPEZLUIS451Ø9567851AV.REAL
```

Probablemente no tenga ninguna dificultad en identificar los nombres de las dos personas, pero su Spectrum no está tan familiarizado como usted con los apellidos comunes. Incluso probablemente no se haya dado cuenta de que los números a continuación de cada nombre corresponden a la edad, número de archivo, número de teléfono y número de la calle de la persona en cuestión. ¿Cómo podrá el programa identificar cada uno de estos datos si cualquiera de ellos puede variar en el futuro?

La respuesta a tales problemas generalmente está proporcionada por una combinación de indicadores y punteros. Los indicadores son señales que se colocan en el bloque principal de datos y que permite al programa identificar la longitud de los elementos que constituyen un registro. Por regla general, los punteros se colocan fuera del bloque principal de datos. Consisten en una lista de las posiciones en la memoria de todos los registros, lo que permite al programa que pueda saltar al medio de un largo y complejo archivo y encontrar, sin cometer ningún error, el primer carácter del registro deseado.

El programa que viene a continuación utiliza indicadores y punteros para manejar un archivo constituido por 28ØØØ caracteres, que constituyen quizá centenares de registros separados, agrupados de una forma aparentemente aleatoria. El programa se llama Archivo. Espero que sea de gran utilidad en su biblioteca de programas. Lo que es muy importante es que las técnicas utilizadas son herramientas esenciales para aquellos que quieran colocar en sus Spectrums, 16 K o 48 K, la máxima cantidad de información. El programa está dimensionado para utilizarlo con la versión de 48 K.

```
1000> PAPER 7: CLS : BORDER 7: I
NK 6: PAPER 0: PRINT PAPER 2;"
        ARCHIVO                 "
1010 PRINT '"FUNCIONES DISPONIBL
ES:"
1020 PRINT '"        1)CREAR NUEVO
ARCHIVO"
1030 PRINT '"        2)ENTRAR INFOR
MACION"
1040 PRINT '"        3)BUSCAR/VISUA
LIZAR/CAMBIAR"
1050 PRINT '"        4)FINALIZAR"
1060 PRINT ''"ENTRE POR FAVOR LA
 QUE REQUIERA."
1070 INPUT Z$
1080 CLS
1090 IF Z$="1" THEN GO SUB 1210
1100 IF Z$="2" THEN GO SUB 1440
1110 IF Z$="3" THEN GO SUB 2180
1120 IF Z$="4" THEN GO SUB 1150
1130 CLS
1140 GO TO 1000
1150 PRINT AT 10,3; INK 7; PAPER
 2;"SISTEMA DE ARCHIVO CERRADO"
1160 BEEP 2,2
1180 INPUT "Ha entrado nueva inf
ormacion que desee guardar? (S/N
)";Q$: IF Q$="N" THEN STOP
1190 SAVE "ARCHIVO": PRINT '"Reb
obine la cinta, luego pulse  cua
lquier tecla para VERIFICAR": PA
USE 0: VERIFY "ARCHIVO": STOP
```

Como regla básica, un programa de utilidad que no empiece con un menú bien claro de las funciones disponibles, es un mal programa. Si no está de acuerdo con esto ahora, ciertamente lo estará algún día, cuando tenga que utilizar de nuevo un programa complejo, pero útil, que no ha sido utilizado durante varias semanas y se encuentre con que tiene que pasar horas y horas buscando a través del listado para intentar recordar qué es lo que hace y cómo.

En este módulo se le pide al usuario que elija entre cuatro funciones numeradas. No se hace ningún intento para eliminar las entradas incorrectas. Los errores en este punto no son importantes. Si se entra un número fuera del rango del 1 al 4, el programa lo ignorará. El módulo también contiene, como una de las cuatro posibles elecciones del menú, la función de STOP. Esto sirve para marcar el final de la utilización del programa y para recordar al usuario que «regrabe» cualquier nuevo dato que haya sido entrado.

Comentario

Línea 1000: Cualquier programa en el Spectrum que no vaya a ser en blanco y negro necesita declarar en algún lugar, cerca del principio, los colores a utilizar para:

a) El borde alrededor de la pantalla.
b) La pantalla.
c) La tinta que se utilizará para los caracteres que aparecen en la pantalla.

En esta línea hay tres instrucciones PAPER separadas. La primera instrucción va sola y, junto con la orden CLS, deja la pantalla de color blanco. En la segunda instrucción, el color del papel se pone a negro para que el menú resalte claramente sobre el fondo blanco. La tercera instrucción PAPER está ligada a una instrucción PRINT. No deja ninguna diferencia permanente con respecto al color del papel, pero nos asegura que la palabra Archivo se visualiza sobre un fondo rojo.

Es importante que vea la diferencia entre las órdenes de color que van solas y seleccionan un color; y aquellas órdenes de color que se refieren únicamente a la sentencia PRINT a la cual están ligadas. En una sentencia PRINT puede seleccionarse cada una de las características mediante estas órdenes asociadas. Por ejemplo:

```
PRINT FLASH 1; OVER 1; INVERSE 1; PAPER 7; INK Ø; ''HOLA''
```

Ninguna de estas órdenes de color producirán efecto sobre las sentencias PRINT situadas en otras partes del programa.

Línea 1Ø7Ø: Cuando tenga una entrada (INPUT) para el menú de un programa, acuérdese de comprobar que las variables utilizadas no estén duplicadas en otras partes del programa.

Línea 1Ø8Ø: El menú se visualiza sobre papel blanco mediante tiras negras gruesas. Pero la última orden PAPER selecciona el negro como color de fondo, por lo que esta orden CLS deja la pantalla totalmente en negro. Seguirá en negro hasta que el programa vuelva al menú.

Línea 119Ø: Para cualquier programa de almacenaje de datos es mucho más conveniente tener una instrucción SAVE dentro del programa que tener que entrarla en modo directo cada vez que se añade un nuevo dato.

Comprobación del módulo 1.1.1

Para comprobar este módulo tan sólo hay que ejecutarlo (RUN) y entrar números dentro del rango del 1 al 3. Entonces el programa deberá detenerse con el mensaje Ø OK, seguido por el correspondiente número de línea, comprendido entre 1Ø9Ø-111Ø. Si se entra un 4 dará como resultado un mensaje para guardar (SAVE) y después

verificar (VERIFY) el programa. Cualquier otra entrada deberá ser ignorada.

MODULO 1.1.2

```
2750 REM ************************
2760 REM SUBRUTINAS FUNCIONALES
2770 REM ************************
2780 INPUT Q$
2790 LET Q$=CHR$ (LEN Q$+1)+Q$
2800 RETURN
2810 PRINT A$(I,2 TO CODE A$(I,1
));":";
2820 RETURN
2830 PRINT FN A$() (2 TO )
2840 RETURN
2850 FOR I=1 TO X
2860 GO SUB 2810
2870 GO SUB 2830
2880 LET C=C+CODE B$(C)
2890 NEXT I
2900 RETURN
```

Este sencillo módulo contiene varias rutinas muy breves que es más económico colocarlas en una subrutina que escribirlas por completo cada vez que se necesiten. Obsérvese la similitud, en este caso, con la utilización de una función definida por el usuario que sirve también para ahorrar espacio de una forma similar. Si una función tiene que trabajar siempre con las mismas variables, entonces una subrutina de una línea puede ser igualmente efectiva. Las funciones definidas por el usuario deben utilizarse cuando la misma función debe trabajar con variables distintas en lugares distintos.

Por ejemplo, la línea 279ø podría sustituirse por una función definida tal como DEF FN Q$ ()=CHR$ (LEN Q$+1)+Q$. Sin embargo, para llamar a esta función se necesitarían siempre dos líneas, INPUT Q$ y LET Q$=FN Q$ () por lo que no se produciría un ahorro real comparado con la única línea necesaria para llamar a la breve subrutina de la línea 278ø. Si tuviésemos tres o cuatro textos distintos sobre los que quisiéramos aplicar esta función, la podríamos definir como DEF FN Q$ (Q$)=CHR$ (LEN Q$+1)+Q$.

Ahora, la función podría aplicarse a otras variables alfanuméricas, colocando simplemente entre paréntesis la variable requerida cuando se llamase a la función. Por ejemplo LET C$=FN Q$(C$).

Si quisiéramos trabajar con C$ con una subrutina de una línea, entonces necesitaríamos otra subrutina para tratar con C$.

La moraleja de esto es sencillamente que el definir funciones sólo por el placer de hacerlo puede ser un gasto de tiempo innecesario. Guarde las funciones definidas para aquellas operaciones que puedan aplicarse a variables distintas en lugares distintos.

El módulo está constituido por cuatro subrutinas, de la siguiente forma:

1) Líneas 278Ø-28ØØ. Esta sección añade a la entrada Q$ el indicador que se mencionó en la introducción. El indicador consta de un único carácter. Recuerde que, en el Spectrum, cada carácter tiene un único código numérico; la lista de estos valores puede encontrarse en el apéndice A del manual del Spectrum. La función CHR$ puede utilizarse para seleccionar el carácter correcto que se corresponde con cualquier valor comprendido entre Ø y 255, mientras que la función CODE convierte cualquier carácter en su valor correspondiente entre Ø y 255.

   Utilizando estas dos funciones pueden guardarse valores comprendidos entre Ø y 255 en un único carácter. En el caso de nuestros indicadores, el único carácter añadido guarda la longitud del texto más uno, que corresponde al propio indicador, de forma que cuando el texto se coloque en el archivo principal de datos, puede utilizarse el indicador para identificar qué trozo de lo que viene después del indicador forma parte del mismo elemento. Si el indicador tiene el valor 11, entonces el elemento consistirá en el indicador y los 1Ø caracteres siguientes.
2) Líneas 281Ø-282Ø. Estas líneas visualizan nombres de elementos tales como nombres y dirección. Obsérvese que el valor del indicador se utiliza aquí para extraer la parte útil de una línea de una tabla.

   Los nombres de elemento están guardados en A$, cuyas líneas tienen 2Ø caracteres de largo. La diferencia entre la longitud del nombre del elemento y la longitud de la línea de la tabla está constituida por espacios que no queremos visualizar.

   La línea 281Ø visualiza únicamente esta parte de la línea que nos interesa de A$ y que contiene los caracteres del nombre del elemento. Ni el indicador ni los espacios se visualizan. Esto puede sernos siempre de ayuda para formatear, cuando un texto se guarda en tablas que son más largas que el propio texto.
3) Líneas 283Ø-284Ø. FN A$ extrae un elemento del archivo principal de datos y se explicará posteriormente en el próximo módulo.
4) Líneas 285Ø-29ØØ. Esta subrutina se utiliza para visualizar registros del archivo. Las variables utilizadas se explicarán en el análisis de los otros módulos.

Comprobación del módulo 1.1.2

El correcto funcionamiento de estas subrutinas tan sólo puede comprobarse de una forma efectiva cuando se hayan entrado los otros módulos.

MODULO 1.1.3

```
1200>REM ************************
1210 REM ESTRUCTURA DE ENTRADA
1220 REM ************************
1230 PRINT PAPER 2;"        ESTRUCT
URA DEL ARCHIVO        "
1240 PRINT '"CUANTOS ELEMENTOS P
OR REGISTRO?"
1250 INPUT X
1260 CLS
1270 DIM A$(X,20)
1280 PRINT PAPER 2;"     NOMBRES D
E LOS ELEMENTOS        "
1290 FOR I=1 TO X
1300 PRINT "ELEMENTO ";I;":";
1310 GO SUB 2780
1320 PRINT Q$(2 TO )
1330 LET A$(I)=Q$
1340 NEXT I
1350 DIM B$(28000)
1360 LET B$(1 TO 4)=CHR$ 2+CHR$
0+CHR$ 2+CHR$ 255
1370 DEF FN A()=256*CODE Y$(2*S-
1)+CODE Y$(2*S)
1380 DEF FN A$()=B$(C TO C+CODE
B$(C)-1)
1390 LET P=5
1400 LET Y$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 3
1410 LET N=2
1420 RETURN
```

Este es el módulo que permite al ARCHIVO asumir formas distintas de acuerdo con los deseos del usuario. A lo largo de este módulo, las tablas y variables principales se preparan para los datos que deben entrarse. Obsérvese que como resultado de esto se perderá cualquier dato que estuviese guardado con anterioridad. No analizaremos con detalle la utilización de las distintas tablas, ya que preferimos dejarlo para cuando empecemos a utilizarlas.

Comentario

Líneas 1230-1340. Un registro típico para el archivo podría consistir en nombre, dirección, edad y número de teléfono. A través de estas líneas el programa guardará en la variable X cuantos de tales elementos haya en cada registro. Se piden los nombres de los ele-

mentos y se guardan en la tabla A$, después de haber colocado un indicador mediante la subrutina de la línea 278Ø. Obsérvese que visualizamos Q$ tras eliminar su primer carácter, ya que el indicador es un carácter sin significado.

En la línea 135Ø se define la tabla principal en la que se guardarán los registros.

En la línea 136Ø se definen dos registros ficticios que marcarán el principio y el final del archivo.

Líneas 137Ø-138Ø. Son dos ejemplos de funciones definidas por el usuario que podrían perfectamente sustituirse por subrutinas de una línea. La primera función extrae el valor de un puntero y se explicará en el módulo 5. La segunda función extrae un elemento del archivo principal, basándose en el valor del indicador que se encuentra en la posición C del archivo.

Línea 139Ø. P es la variable que se utiliza para guardar la posición del primer espacio vacío de B$. B$ tendrá siempre 28ØØØ caracteres de largo, pero tan sólo utilizaremos una parte del mismo. Evidentemente, necesitamos saber cuánto se ha utilizado ya.

Línea 14ØØ. Y$ guarda los punteros en la forma de códigos de carácter, un método que se analiza en relación con el módulo 5.

Línea 141Ø. N es la variable que guarda el número de registros del archivo.

Comprobación del módulo 1.1.3

Ahora ya podemos comprobar los módulos 2 y 3. Ejecute el programa y seleccione la función 1 del menú. Debe especificar un número de elementos y después darles nombres. Una vez hecho esto, detenga el programa y, en modo directo, visualice las distintas tablas y variables así:

B$:??? COPY
Y$:????
N:2
P:5

X debe ser igual al número de elementos que haya especificado y la tabla A$ debe tener X líneas, cada una con un nombre de elemento con un indicador colocado delante del mismo.

MODULO 1.1.4

```
1430>REM ************************
1440 REM ENTRADA NORMAL
1450 REM ************************
```

```
1460 LET R$=""
1470 PRINT PAPER 2;"           R
EGISTROS            "
1480 PRINT '"COMANDOS DISPONIBLE
S:"
1490 PRINT '">ENTRAR ELEMENTO ES
PECIFICADO"''">""ZZZ"" PARA SALI
R"
1500 PRINT "********************
************"
1510 PRINT "LONGITUD ARCHIVO:";P
-1;"/";LEN B$
1520 FOR I=1 TO X
1530 GO SUB 2810
1540 GO SUB 2780
1580 PRINT Q$(2 TO )
1590 IF Q$(2 TO )="ZZZ" THEN RET
URN
1600 LET R$=R$+Q$
1610 NEXT I
1620 CLS
1630 GO SUB 1660
1640 GO TO 1440
```

El propósito de este módulo es aceptar la entrada de un registro, compuesto del número correcto de elemento y presentar este registro a la sección del programa que lo insertará en su lugar correcto del archivo.

Comentario

Línea 16ØØ. R$ es el registro y está compuesto por cierto número de Q$ sucesivos que se han juntado.

Comprobación del módulo 1.1.4

Si ya ha entrado algunos nombres de elemento correctos, entonces empiece el programa con GOTO 1 y llame a la función 2 del menú. Se le pedirá una entrada para cada nombre de elemento. Tras la aparición del número correcto de nombres de elemento el programa se detendrá con el mensaje Ø OK, 163Ø:1. El tamaño dèl archivo deberá ser 4/28ØØØ y, si visualiza R$ éste deberá consistir en los elementos entrados, cada uno de ellos precedido por un carácter indicador.

MODULO 1.1.5

```
1650>REM ***********************
1660 REM PONER DATOS EN ARCHIVO
1670 REM ***********************
1680 IF P+LEN R$-1<LEN B$ THEN G
O TO 1730
```

```
1690 PRINT AT 14,10;"ARCHIVO LLE
NO"
1700 PRINT ''"Pulsar cualquier t
ecla para"'"continuar"
1710 PAUSE 0
1720 RETURN
1730 LET POTEN=INT (LN (N-1)/LN
2)
1740 LET S=2↑POTEN
1750 LET T$=R$(2 TO CODE R$(1))
1760 FOR K=POTEN-1 TO 0 STEP -1
1770 LET C=FN A()
1780 LET U$=FN A$() (2 TO )
1790 LET S=S+(2↑K)*(T$>U$)-(2↑K)
*(T$<U$)
1810 IF S>N-1 THEN LET S=N-1
1820 IF S<2 THEN LET S=2
1830 NEXT K
1840 LET C=FN A()
1850 LET U$=FN A$() (2 TO )
1860 IF T$<U$ THEN LET S=S-1
1870 LET B$(P TO P+LEN R$-1)=R$
1880 LET N=N+1
1890 LET Y$=Y$(1 TO 2*S)+CHR$ IN
T (P/256)+CHR$ (P-256*INT (P/256
))+Y$(2*(S+1)-1 TO )
1900 LET P=P+LEN R$
1910 RETURN
```

Este módulo es el más complejo del programa. Antes de pasar a un comentario detallado discutiremos dos puntos:

1) La utilización de textos para guardar números.
2) La técnica de la búsqueda binaria.

Números en textos

Ya hemos visto, en el módulo 3, que los punteros de nuestro programa se guardan en una variable alfanumérica, Y$. Quizá se pregunte por qué los valores numéricos no se guardan directamente en una tabla numérica. La respuesta radica tanto en el ahorro de memoria como en el ahorro de tiempo, aunque este último es el aspecto más significativo. Veamos primero el ahorro de memoria.

Para poder tratar el número máximo de registros que puedan entrarse, una tabla numérica para los punteros tendría que declararse con alrededor de 2ØØØ elementos. Es muy poco probable que tenga que tratar dos mil registros, pero es posible. Se encontraría con serios problemas si la tabla no tuviese suficientes espacios para el número de registros a los que debe apuntar. Una tabla no puede redimensionarse sin perder todo lo que hay en ella. El verdadero problema es que una tabla numérica de dos mil elementos, debido a la forma en que el BASIC de Sinclair guarda los números, ocuparía unos 1ØØØØ bytes de memoria. Esto es una gran proporción del total de la memoria disponible.

El Spectrum asigna cinco bytes de memoria a cada número guardado en una tabla, en un intento de cubrir el mayor rango posible de números; de hecho hasta 4.294.967.295. En nuestro caso no necesitamos un número tan grande: nuestro archivo tiene tan sólo 28ØØØ caracteres de largo, por lo que tan sólo necesitamos números enteros desde el 1 al 28ØØØ. Utilizando textos de dos caracteres se pueden representar estos números.

Cada carácter tiene un código numérico asociado único, comprendido entre Ø y 255. Un único carácter puede utilizarse para guardar cualquier valor comprendido entre Ø y 255 utilizando simplemente el carácter que tiene este código. Así, el carácter A representa al número 65 y la palabra clave GOTO —que es tan sólo otro carácter en lo que se refiere al Spectrum— representa al número 236. Los números mayores que 255 se representan utilizando un segundo carácter para guardar el número de veces que contiene 255, de forma parecida en la que 3 en el número 36 representa 3 veces 1Ø en nuestro sistema decimal. Por lo tanto, dos caracteres nos dan la posibilidad de guardar cualquier número entero positivo hasta el 255*256+255. Esto es igual a 65535 y es más que suficiente para nuestro archivo de 28ØØØ caracteres.

Suponiendo que se desee utilizar tan sólo números enteros positivos comprendidos entre Ø y 65535, pueden ahorrarse tres de los cinco bytes que el Spectrum utilizaría si se guardasen los mismos números en una tabla numérica.

Lo malo es que dos de los tres bytes ahorrados se desperdician después para conseguir velocidad.

Imagínese de nuevo nuestra tabla numérica de 2ØØØ elementos e imagínese que quiere añadir o borrar un número que esté colocado en algún lugar cerca del principio del mismo. Si borra sencillamente un valor indeseado, quedará un agujero, o mejor dicho un cero, en el lugar donde estaba antes el número. Si inserta un número lo escribirá encima de lo que ya había allí. Para evitar cualquiera de estos resultados indeseados tendrá que asegurarse de que cada uno de los elementos de la tabla puede desplazarse un lugar hacia arriba o hacia abajo. Si la posición en la que desea insertar un nuevo número es la posición 1, entonces tendrán que desplazarse 1999 números para dejar espacio. Puede realizarse mediante 3 líneas de BASIC que formen un simple bucle, pero esto necesita tiempo, especialmente en un Spectrum. Ni sus mejores amigos describen al Spectrum como terriblemente rápido.

Ahora compare este bucle, que repite la operación 1999 veces, con esto:

```
LET A$="XX"+A$
```

Utilizando las grandes posibilidades del Spectrum para el manejo de textos podemos insertar dos bytes al principio, o al final, o en el medio, de un texto con una sola instrucción. Esto es muy rápido, pero tiene un inconveniente; dobla momentáneamente la cantidad de espacio utilizado por el texto. El Spectrum necesita guardar en su memoria, incluso aunque sea sólo por un momento, en nuevo A$ que la línea está creando, junto con el antiguo A$ que se está utilizando para construirlo. Esta limitación es uno de los mayores inconvenientes para el manejo de textos del Spectrum y es difícil de evitar. Significa que Y$, que se utiliza para guardar los pares de caracteres que utilizamos como punteros para los registros del archivo principal, es efectivamente el doble de largo de lo que parece, ya que, en nuestro intento de aumentar la velocidad, se doblará momentáneamente cada vez que añadamos o borremos algo del mismo. Este doblaje quizá sea tan sólo momentáneo, pero no obstante tendremos que reservar espacio de memoria para el mismo. Es una pena, pero tendremos que aprender a vivir con esto.

Este inconveniente es el motivo por el que no utilizamos el mismo método para insertar o borrar datos en nuestro archivo principal B$. El hacerlo representaría reducir a la mitad la cantidad de espacio que podría utilizarse para los registros. Hemos declarado B$ como una tabla de longitud fija y, cuando queramos borrar algo, moveremos el resto del archivo hacia abajo, registro a registro, para rellenar el hueco creado.

Búsqueda binaria

Utilizaremos la técnica de la búsqueda binaria para reducir el número de comparaciones posibles que hay que realizar para encontrar el lugar correcto donde hay que insertar un nuevo registro, desde 125ØØ a 15 comparaciones posibles. Considérese el siguiente ejemplo:

Hemos establecido un archivo que contiene 2ØØØ elementos y hay un nuevo elemento que debe ser insertado en la posición 1731, aunque el programa no lo ha descubierto todavía. El programa empieza su búsqueda, buscando en el primer registro del archivo y comparándolo con el nuevo registro que debe insertarse. Encuentra que el nuevo registro es mayor que éste por orden alfabético, por lo que el programa continúa examinando el siguiente registro del archivo. Al final, tras realizar 1731 comparaciones, el programa encuentra el primer registro del archivo que es mayor que el nuevo registro. Ya habrá encontrado la posición correcta para el nuevo elemento.

Compare este proceso directo con el siguiente, para un archivo del mismo tamaño y una inserción en la misma posición.

El programa empieza examinando el registro que ocupa la posición 1ø24, ya que 1ø24 es la mayor potencia de 2 que es menor o igual que el número de registros del archivo. El programa encuentra que el registro 1ø24 es menor por orden alfabético que el nuevo registro, por lo que sumará 1ø24/2 al 1ø24 original, que da como resultado 1536. El registro situado en la posición 1536 sigue siendo menor que el nuevo registro, por lo que suma 1ø24/4 a 1526, que da 1792. El registro situado en 1792 es mayor que el nuevo registro por lo que se restará 1ø24/8 de 1792 dando 1664. La búsqueda sigue en las siguientes posiciones del archivo, realizándose las siguientes sumas o restas.

1644 (después suma 64)
1728 (después suma 32)
176ø (después suma 16)
1744 (después suma 8)
1736 (después suma 4)
1732 (después suma 2)
173ø (después suma 1)
Resultado final: 1731

La potencia de la búsqueda binaria es pues evidente.

Comentario

Líneas 168ø-172ø. Estas líneas comprueban si queda espacio suficiente en el archivo para el nuevo registro.

Líneas 173ø-183ø. Se aplica la búsqueda binaria a los registros de B$. La búsqueda se realiza con respecto al orden alfabético del primer elemento de cada registro. Para obtener una explicación de cómo entiende el Spectrum el orden alfabético, ver la página 95 del manual del Spectrum.

Línea 173ø. Encuentra la potencia más alta de 2 que es menor o igual que el número de registros del archivo. Utiliza la función logaritmo. La posición de búsqueda se pone igual a este valor.

Línea 175ø. En T$ se coloca el primer elemento del registro correspondiente a la posición de búsqueda.

Líneas 176ø-183ø. Este bucle suma o resta potencias de 2 de acuerdo con los principios establecidos en el análisis de la clasificación binaria.

Línea 177ø. FN A fue definida en la línea 137ø. Extrae de dos caracteres contenidos en Y$ un valor numérico que corresponde al puntero del primer carácter de un registro del archivo principal.

Línea 178ø. FN A$ fue definida en la línea 138ø. Extrae del ar-

chivo principal el elemento cuyo indicador se encuentra en la posición C de B$.

Línea 179Ø. Esta línea necesita más explicación. Una condición como T$>U$ o es verdadera o es falsa, pero en el uso habitual no puede decirse que tenga un valor de la misma forma que un número o una variable lo tiene. Sin embargo, para el Spectrum, T$>U$ tiene un valor real que es 1 si la condición es verdad, o Ø si la condición es falsa. El valor de la condición puede utilizarse en un programa de la misma forma que se utilizaría un número o una variable. En esta línea en particular, si T$>U$, la condición tendrá el valor 1 y se sumara (2^K)*1 al valor contenido en S. Por otra parte si T$<U$, la condición será Ø y se restará (2^K)*Ø del valor contenido en S. Si T$ es menor que U$ entonces se invertirán los papeles, mientras que si T$ fuese igual a U$ entonces ambas condiciones serían falsas y S quedaría inalterado.

Líneas 181Ø-182Ø. Si S, que es la posición de búsqueda, señala a uno de los registros ficticios, estas dos líneas vuelven de nuevo al bloque principal de datos.

Líneas 184Ø-185Ø. Una vez completa la búsqueda binaria, el elemento que ocupa la posición seleccionada se extrae para su examen. Si el elemento en esta posición y el nuevo elemento son iguales, el nuevo elemento se coloca después del elemento existente. Si no son iguales, entonces el nuevo elemento se coloca antes del elemento existente.

Línea 187Ø. El nuevo registro se añade al final del archivo. El orden correcto de los registros del archivo se mantiene únicamente en Y$. Con tal de que Y$ sepa donde está el registro número 378, por ejemplo, no tiene importancia si está realmente guardado en la posición 378.

Comprobación del módulo 1.1.5

Es difícil comprobar este módulo hasta que no se hayan añadido al programa las funciones de búsqueda y visualización, que permitan visualizar fácilmente los registros. Si quiere puede entrar algunos registros y después detener el programa para comprobar si han sido insertados en B$. Recuerde que se habrán insertado en el orden en que hayan sido entrados. Si quiere, también puede examinar Y$ mediante este bucle.

```
9ØØØ FOR S=1 TO LEN Y$ STEP 2:PRINT FN A ():NEXT S
```

Con esto se visualizarán los valores de los punteros, que podrá asociar con los principios de los registros del archivo principal.

```
2170>REM ************************
2180 REM BUSQUEDA
2190 REM ************************
2200 LET S=2
2210 PRINT PAPER 2;"
BUSQUEDA                  "
2220 PRINT ''"COMANDOS DISPONIBL
ES:"
2230 PRINT ">ENTRAR ELEMENTO PAR
A LA"'" BUSQUEDA NORMAL"'">PRECE
DER CON ""SSS"" PARA LA"'" BUSQU
EDA ESPECIAL"'">PRECEDER CON ""I
II"" PARA BUSCAR"'" POR EL PRIME
R CARACTER DEL"'" REGISTRO"'">""
ENTER"" PARA OBTENER EL PRIMER"'
" ELEMENTO DEL ARCHIVO"
2240 PRINT "********************
************"
2250 PRINT ''"ENTRAR ELEMENTO A
BUSCAR:";
2260 GO SUB 2780
2270 PRINT Q$(2 TO )
2280 LET S$=Q$
2290 IF LEN S$=1 THEN GO TO 2510
2300 LET C=FN A()
2310 IF LEN S$<5 THEN GO TO 2430
2320 IF S$(2 TO 4)<>"III" THEN G
O TO 2390
2330 FOR I=S TO N
2340 LET S=I
2350 LET C=FN A()
2360 IF B$(C+1)=S$(5) THEN GO TO
 2510
2370 NEXT I
2380 RETURN
2390 IF S$(2 TO 4)<>"SSS" THEN G
O TO 2430
2400 GO SUB 2920
2410 IF C4=1 THEN GO TO 2510
2420 RETURN
2430 FOR I=1 TO X
2440 IF FN A$()=S$ THEN GO TO 25
10
2450 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2460 LET C=C+CODE B$(C)
2470 NEXT I
2480 LET S=S+1
2490 LET C=FN A()
2500 GO TO 2430
2510 LET C=FN A()
2520 LET C4=0
2530 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2540 CLS
2550 PRINT "ENTRADA ";S-1;":-"
2560 GO SUB 2850
2570 LET S=S+1
2580 PRINT AT 14,0; PAPER 2;"
         BUSQUEDA              "
2590 PRINT "COMANDOS DISPONIBLES
:"
2600 PRINT ">""ENTER"" PARA VISU
ALIZAR EL"'" SIGUIENTE ELEMENTO"
'">""ZZZ"" PARA ABANDONAR LA FUN
CION"'">""AAA"" PARA MODIFICAR"'
">""CCC"" PARA CONTINUAR LA"'" B
USQUEDA"
2610 INPUT P$
```

```
2620 CLS
2630 IF P$="CCC" THEN GO TO 2300
2640 IF P$="" THEN GO TO 2510
2650 IF P$<>"AAA" THEN GO TO 271
0
2660 LET C=FN A()
2670 CLS
2680 GO SUB 1930
2710 IF P$="ZZZ" THEN RETURN
2720 IF P$="AAA" THEN RETURN
2730 CLS
2740 GO TO 2260
```

El propósito de este módulo es visualizar los registros del archivo, sea uno a uno desde el principio o empezando con el primer registro que satisfaga ciertas condiciones de búsqueda. Una vez visualizado un registro, el módulo da la posibilidad al usuario de elegir entre continuar la búsqueda, examinar el siguiente registro, cambiar el registro o borrarlo del archivo. Obsérvese la continua utilización de FN A y de FN A$ para obtener la dirección de un registro y extraerlo del archivo.

Comentario

Línea 22ØØ. S es el número del registro que se está examinando en la actualidad. Inicialmente se pone a 2 ya que el primer registro del archivo es en realidad un registro ficticio.

Líneas 229Ø-238Ø. Si el usuario entra una instrucción de búsqueda que empiece con III, el programa recorre el primer elemento de cada registro hasta que encuentre uno que empiece con el carácter entrado a continuación de III. Si no se encuentra tal elemento, el programa vuelve al menú principal.

Líneas 239Ø-242Ø. La búsqueda especial, que busca cualquier combinación especificada de caracteres, independientemente de si es un elemento completo o no, es llevada a cabo por una subrutina separada que es llamada desde estas líneas, si la instrucción de búsqueda empieza con SSS.

Líneas 243Ø-25ØØ. Se examinan los elementos completos del archivo para ver si coinciden con el elemento que se ha pedido que busque el programa. Esto es mucho más rápido que la búsqueda especial, que recorre el archivo carácter a carácter. La rápida búsqueda binaria no puede utilizarse ya que tan sólo los primeros elementos de cada registro están por orden alfabético. Para que esta búsqueda considere encontrado el elemento entrado, éste debe ser exactamente el mismo que el elemento de la memoria. Si buscamos Sánchez, J en el archivo, no encontrará Sánchez, Juan a menos que utilicemos la búsqueda especial, en cuyo caso SSSSánchez, J encontraría Sánchez, J o Sánchez, Juan, pero sería mucho más lento.

Líneas 251Ø-257Ø. Esta sección visualiza un registro utilizando la subrutina de la línea 285Ø que ya hemos examinado.

Líneas 258Ø-274Ø. Una vez descubierto un registro que satisface los criterios de búsqueda, el módulo ofrece ahora al usuario la posibilidad de recorrer el archivo registro a registro, buscando el siguiente registro que satisfaga el criterio de búsqueda original o llamar a la rutina que permite modificar o borrar el registro.

Comprobación del módulo 1.1.6

Puede comprobar el funcionamiento correcto de todas las funciones de búsqueda excepto la búsqueda especial. La función de modificación de un registro todavía no ha sido entrada.

MODULO 1.1.7

```
2910>REM ************************
2920 REM BUSQUEDA ESPECIAL
2930 REM ************************
2940 LET C4=0
2950 FOR H=S TO N-1
2960 LET S=H
2970 LET C=FN A()
2980 LET C1=C
2990 FOR I=1 TO X
3000 LET C1=C1+CODE B$(C1)
3010 NEXT I
3020 FOR J=C+1 TO C1-LEN S$+5
3030 IF B$(J TO J+LEN S$-5)<>S$(
5 TO ) THEN GO TO 3060
3040 LET C4=1
3050 RETURN
3060 NEXT J
3070 NEXT H
3080 LET C4=0
3090 RETURN
```

Este módulo contiene la rutina de búsqueda especial mencionada anteriormente.

Comentario

Línea 294Ø. C4 es el indicador utilizado para señalar, al volver del módulo 6, si la combinación de caracteres especificada se ha encontrado.

Líneas 298Ø-3Ø1Ø. C1 se pone inicialmente igual a la dirección inicial del registro bajo examen. A C1 se le suman los indicadores asociados a los X elementos del registro, con lo que C1 será ahora

igual a la dirección inicial del siguiente registro. Obsérvese que ahora estamos hablando de la dirección inicial del siguiente registro en el archivo principal, y no del siguiente registro por orden alfabético.

Líneas 3020-3060. El registro se examina carácter a carácter, buscando la coincidencia con la combinación de caracteres especificada en la instrucción de búsqueda.

Comprobación del módulo 1.1.7

Entre una serie de combinaciones de caracteres, algunas de las cuales estén en el archivo y otras que no lo estén. No se olvide de poner delante SSS.

MODULO 1.1.8

```
1920>REM ************************
1930 REM CAMBIAR REGISTRO
1940 REM ************************
1950 LET S=S-1
1960 LET C=FN A()
1970 LET R$=""
1980 PRINT "REGISTRO ";S-1;":-"
1990 FOR I=1 TO X
2000 GO SUB 2810
2010 GO SUB 2830
2020 PRINT AT 17,0; PAPER 2;"
      MODIFICAR                  "
2030 PRINT "COMANDOS DISPONIBLES
:"
2040 PRINT ">""ENTER"" NO MODIFI
CAR"'">""ZZZ"" ELIMINA TODO EL R
EGISTRO"'">ENTRAR NUEVO ELEMENTO
"
2050 GO SUB 2780
2060 IF LEN Q$=1 THEN LET R$=R$+
B$(C TO C+CODE B$(C)-1)
2070 LET C=C+CODE B$(C)
2080 CLS
2090 IF LEN Q$=1 THEN GO TO 2120
2100 IF Q$(2 TO )="ZZZ" THEN GO
TO 2130
2110 LET R$=R$+Q$
2120 NEXT I
2130 GO SUB 3130
2140 IF Q$(2 TO )="ZZZ" THEN RET
URN
2150 GO SUB 1660
2160 RETURN
```

Este módulo da al usuario la opción de cambiar o borrar el registro presentado al módulo por la función de búsqueda.

Comentario

Líneas 2Ø5Ø-213Ø. Quizá recuerde que en el módulo 4 los nuevos registros se construían sobre R$. En estas líneas se crea un R$ modificado, que está constituido o bien por elementos tomados directamente del registro del archivo, o por elementos entrados para sustituir a los originales. Después se borra el registro original del archivo, llamando a la subrutina de la línea 313Ø.

Línea 214Ø. Si el usuario no ha especificado que el registro deba borrarse, el registro modificado, contenido en R$, es presentado al módulo 5 que lo insertará en el lugar correcto.

Comprobación del módulo 1.1.8

La comprobación total de este módulo debe esperar la entrada del siguiente, aunque puede comprobar que el módulo visualiza realmente el registro seleccionado, elemento a elemento y que cualquier cambio entrado queda registrado en R$. Tras visualizar todos los elementos del registro, el programa se detendrá con el mensaje Ø OK, 213Ø:1.

MODULO 1.1.9

```
3100>REM ************************
3110 REM DESPLAZAR ARCHIVO
3120 REM ************************
3130 LET C=FN A()
3140 LET DESPL=1000
3150 LET C1=C
3160 LET C3=C
3170 FOR I=1 TO X
3180 LET C1=C1+CODE B$(C1)
3190 NEXT I
3200 LET C2=C1-C
3210 FOR I=C1 TO LEN B$-1 STEP D
ESPL
3220 IF LEN B$-I+1<DESPL THEN LE
T DESPL=LEN B$-I+1
3230 LET S$=B$(I TO I+DESPL-1)
3240 LET B$(C TO C+DESPL-1)=S$
3250 LET C=C+DESPL
3260 NEXT I
3270 LET Y$=Y$(1 TO 2*(S-1))+Y$(
2*(S+1)-1 TO )
3280 FOR I=1 TO N-1
3290 LET S=I
3300 LET C=FN A()
3310 IF C<=C3 THEN GO TO 3350
3320 LET C=C-C2
3330 LET Y$(2*I-1)=CHR$ INT (C/2
56)
3340 LET Y$(2*I)=CHR$ (C-256*INT
 (C/256))
3350 NEXT I
```

```
3360 LET P=P-C2
3370 LET N=N-1
3380 RETURN
```

Cuando se borre un registro del archivo, quedará un hueco que deberá rellenarse. La función de este módulo es borrar un registro especificado, desplazando el archivo, sucesivamente sobre sí mismo. El archivo no se mueve registro a registro, sino en bloques de mil caracteres.

Comentario

Línea 317∅. Este bucle coloca C1 en la dirección inicial del siguiente registro.

Líneas 321∅-326∅. Este bucle desplaza un bloque de 1∅∅∅ caracteres del archivo hacia abajo, la longitud correspondiente al registro que hay que borrar, empezando el primer bloque en C1.

Líneas 323∅-324∅. Si B$ ha sido mencionado a ambos lados de una ecuación —por ejemplo LET B$ (C TO C+SHIFT−1)=B$(I TO I+SHIFT−1)— una copia de B$ se crearía momentáneamente y el programa se quedaría sin memoria.

Línea 327∅. Se elimina el puntero del registro borrado.

Líneas 328∅-335∅. Ahora deberán corregirse todos los punteros correspondientes a los registros del archivo que han sido desplazados hacia abajo, ya que sus direcciones iniciales ahora serán distintas.

Comprobación del módulo 1.1.9

Utilice la función de MODIFICAR para borrar uno o dos elementos, después cambie los primeros elementos de algunos registros de forma que tengan que cambiar de sitio dentro del archivo. Después de cada cambio o borrado, utilice la función de BUSQUEDA para comprobar que el archivo sigue todavía en el orden correcto. Si estas comprobaciones son satisfactorias, el programa ya estará completo.

Resumen

Ahora ya ha terminado con la entrada de un programa sustancial y complejo, que espero encuentre de utilidad para varias aplicaciones. Además de esto, en este proceso habrá aprendido varias técnicas que le dejarán en una buena posición, si se decide embarcar en la reali-

zación de programas más ambiciosos que traten sobre el almacenaje y proceso de datos no numéricos.

Ha aprendido como estructurar bloques de datos mediante la utilización de punteros e indicadores. Ha visto como pueden utilizarse las variables alfanuméricas de una forma efectiva para guardar un cierto número de datos numéricos. Tiene un ejemplo funcionando de la potente técnica de búsqueda binaria.

Sin embargo, lo que todavía es más importante, si se ha tomado la molestia de comprender lo que ha estado entrando, habrá ganado confianza sobre el hecho de que los grandes bloques complejos de datos pueden procesarse sin que todo degenere en un caos; después de todo, la mayor parte del arte de la programación es el meterse en aplicaciones que parecen terriblemente complicadas, junto con la perseverancia de seguir la tarea hasta el final.

Posibles mejoras

Si ha comprendido lo que ha entrado, quizá quiera emprender algunas de las siguientes tareas.

1) El programa está deliberadamente escrito sin utilizar demasiado las líneas multisentencia. Una vez que el programa funcione correctamente sería una buena idea intentar reducirlo combinando líneas —aprenderá mucho sobre las ventajas e inconvenientes de las líneas multisentencia.
2) Ya he mencionado anteriormente que en el módulo de búsqueda no se utiliza la búsqueda binaria. ¿Por qué no añadir otra instrucción de búsqueda que se refiera únicamente al primer elemento de cada registro y que utilice la rutina de búsqueda binaria para llevar a cabo la búsqueda?
3) El programa tal como está estructurado no puede tratar archivos o registros que tengan un número variable de elementos. Este tipo de estructura es bastante frecuente, por ejemplo una receta con título, y un número variable de ingredientes e instrucciones. Es relativamente sencillo modificar el programa para que funcione con tres elementos por registro, pero con el segundo elemento subdividido en varios subelementos. La función de MODIFICAR deberá ser capaz de añadir o borrar subelementos.
4) El programa no prevé una salida para la impresora del ZX; esto podría rectificarse fácilmente.

2. Realización del presupuesto. El Spectrum como banquero

En el capítulo anterior hemos examinado las técnicas del manejo de datos no numéricos. En este capítulo trataremos con números y en particular con dinero.

En el capítulo se presentan tres programas. El primero de ellos, Presupuesto, es una herramienta financiera potente y flexible que le permite examinar las consecuencias de decisiones financieras complejas y obtener una imagen del aspecto que tendrán sus finanzas durante los próximos doce meses. A continuación, Contabilidad es un sencillo programa que genera un conjunto de cuentas en un formato fácil de leer para usted. Finalmente, Cuentas Bancarias es un programa que le permite seguir el desarrollo de su cuenta bancaria sin tener que esperar a que el banco le envíe el estado de cuentas.

2.1 Presupuesto

Este es un programa sustancial, considerablemente más largo que Archivo. Esto quizá le parezca sorprendente considerando que la cantidad de información que el programa maneja es considerablemente menor que los 28ØØØ caracteres de Archivo. Hay tres motivos principales para esto. Primeramente, Presupuesto es un programa más flexible en cuanto a lo que permite hacer al usuario. La lista de funciones del programa es mucho más extensa. En segundo lugar, hay una buena cantidad de cálculos repetitivos que hay que realizar y esto utiliza espacio. En tercer lugar está la cuestión del formateado. La información guardada por el programa sería indigerible si no fuera por el hecho de que una gran parte del programa se dedica a asegurar que los datos se presenten de una forma clara.

MODULO 2.1.1

```
3330>REM *************************
3340 REM PREGUNTAS
3350 REM *************************
3360 PRINT AT P1,0;O$
3370 PRINT AT P1,P2;P$
```

```
3375 IF H=2 THEN PRINT AT 21,29;
"(H)"
3380 INPUT Q$
3390 PRINT AT 20,0;O$;O$
3400 PRINT AT 20,0;">>";Q$;"<<"
3410 PRINT AT 21,0;"""ENTER"" pa
ra confirmar"
3420 INPUT R$
3430 PRINT AT 20,0;O$;O$
3440 IF R$="" THEN GO TO 3460
3450 GO TO 3360
3460 LET P2=0
3470 DIM T$(9)
3480 LET T$=Q$
3490 RETURN
```

Este parece un punto extraño para empezar, pero muy pocos de los otros módulos del programa funcionarán hasta que éste no haya sido insertado. El propósito de este módulo es manejar casi todos los mensajes que serán necesarios durante la ejecución del programa. Proporciona un método flexible para formatearlos y un sencillo procedimiento para la comprobación de errores.

Algunos programas llegan a tener una extraordinaria longitud para protegerse contra entradas sin sentido que harían perderse al programa o dar resultados sin significado. Estas técnicas pueden contribuir a la robustez de un programa, pero nunca pueden hacerlo completamente seguro contra todas las contingencias. Tan sólo la experiencia de un programa al utilizarlo puede determinar si vale la pena el esfuerzo extra para incluir tales comprobaciones. En este módulo adoptamos la técnica de recordar sencillamente al usuario lo que se ha entrado y pedirle confirmación antes de que sea aceptado. El motivo de que tengamos una comprobación de error en este programa mientras que no era así en Archivo es que es mucho más fácil el cometer un error cuando se entra una serie de números y no darse cuenta de que se ha cometido este error. Esto es mucho más importante que el escribir mal un nombre al entrarlo en un programa de archivo.

Comentario

Línea 336Ø. P1 es sencillamente un número de línea que se declara antes de llamar a esta rutina. O$ es una línea de 32 blancos y que asegura que la línea en la que va a visualizarse el mensaje está borrada.

Línea 337Ø. P2 es la posición dentro de la línea. Al principio del programa se pone a cero —al principio de la línea— y permanece a cero a menos que se cambie antes de que esta rutina sea llamada. P$ es el texto del mensaje en cuestión.

Líneas 339Ø-345Ø. La entrada se vuelve a visualizar y se pide al usuario que confirme que la entrada es la que pretendía realizar.

Línea 346Ø. En el caso de que P2 haya sido puesto a un valor distinto de cero, será reinicializado.

Líneas 347Ø-348Ø. En algunos casos la entrada será comparada con un texto guardado en una tabla, como es el caso de los nombres de los meses. Estas líneas de la tabla tienen 9 caracteres de largo, independientemente del número de letras que tengan. La forma más fácil de realizar efectivamente esta comparación es convertir también el texto de entrada para que tenga 9 caracteres.

Comprobación del módulo 2.1.1.

Tendrá que entrar en modo directo los valores de O$, P1, P2, y P$. Después un GOTO al módulo, con lo que deberá visualizarse el mensaje P$ en la posición especificada e invitarle a confirmar la entrada.

MODULO 2.1.2

```
1000>REM ************************
         LA PRIMERA VEZ QUE SE
         EJECUTE ESTE PROGRAMA
         ES NECESARIO ENTRAR EN
         MODO DIRECTO:
         "LET MO=X"
         DONDE X ES EL NUMERO DE
         MES EN CURSO
1010 REM ************************
1020 LET H=1
1030 DIM O$(32)
1035 LET K$="[redacted]"
1040 LET L$="*******************
*************"
1050 DATA "ENERO","FEBRERO","MAR
ZO","ABRIL","MAYO","JUNIO","JULI
O","AGOSTO","SETIEMBRE","OCTUBRE
","NOVIEMBRE","DICIEMBRE"
1060 DIM N$(12,9)
1070 RESTORE
1080 FOR I=1 TO 12
1090 READ N$(I)
1100 NEXT I
1180 LET P2=0
1185 CLS
```

Este módulo establece las variables necesarias. Obsérvese la utilización de la función DATA y de sus órdenes asociadas READ y RESTORE.

Comentario

Líneas 1ØØØ-1Ø1Ø. Una vez haya entrado MO por vez primera, tal vez sea mejor dejar esta sentencia REM en su lugar para el caso de que inadvertidamente ejecute el programa otro día y borre MO.

Línea 1Ø2Ø. Esta variable será explicada en el módulo 5.

Líneas 1Ø3Ø-1Ø4Ø. Estos textos se utilizan para formatear la pantalla.

Líneas 1Ø5Ø-11ØØ. En la tabla N$ se colocan los nombres de los meses, utilizando READ, DATA y RESTORE. Obsérvese que el RESTORE es necesario ya que este programa se empieza siempre con un GOTO 1, lo que no reinicializa los punteros de lectura al principio de los datos. Si el programa se ejecuta por segunda vez, el puntero ya habrá alcanzado el final de los datos, con lo que obtendrá un mensaje de error indicando que se ha quedado sin datos.

Comprobación del módulo 2.1.2

Esto se hará mejor cuando se hayan entrado los otros módulos que utilizan las variables.

MODULO 2.1.3

```
3590 REM ***********************
3600 REM REGISTRAR MES
3610 REM ***********************
3615 LET Y=MO+11
3620 PRINT AT 0,5;"PRESUPUESTO D
OMESTICO"
3630 LET P$="A QUE MES ESTAMOS?"
3650 LET P1=3
3660 GO SUB 3360
3670 IF T$=N$(MO) THEN RETURN
3680 FOR I=1 TO 12
3690 IF T$=N$(I) THEN GO TO 3740
3700 NEXT I
3710 CLS
3720 PRINT AT 9,0;"DEBE HABER UN
 ERROR EN EL MES ENTRADO. NO CON
OZCO UN MES LLAMADO ";Q$;"!"
3730 GO TO 3620
3740 LET M2=I
3750 FOR I=MO TO M2-1+12*(M2<MO)
3755 LET I1=I-12*(I>12)
3760 CLS
3770 PRINT AT 0,9;"ACTUALIZACION
"
3780 PRINT ''"ENTRE POR FAVOR LA
S CANTIDADES"'"PARA EL PROXIMO "
;N$(I)
3790 FOR J=1 TO N1
3800 LET P1=5
3830 LET P$=B$(1,J)+": ("+STR$ B(
1,J,I1)+"):"
3840 GO SUB 3360
```

```
3850 LET B(1,J,I1)=VAL Q$
3860 PRINT AT 5,0;O$
3870 NEXT J
3900 LET P$="INGRESO PRINCIPAL: (
"+STR$ E(1,I1)+"):"
3910 GO SUB 3360
3920 LET E(1,I1)=VAL Q$
3950 LET P$="INGRESO ADICIONAL: (
"+STR$ E(3,I1)+"):"
3960 GO SUB 3360
3970 LET E(3,I1)=VAL Q$
3980 NEXT I
3990 LET MO=M2
4000 LET Y=MO+11
4010 GO SUB 2600
4015 GO SUB 2900
4020 RETURN
```

El propósito de este módulo es comprobar si el mes ha cambiado desde que se utilizó el programa por última vez. Si el mes ha cambiado, los datos desactualizados se borran y se piden los nuevos datos. Si el cambio es por ejemplo de mayo a junio, los datos relativos a mayo se borran y se entran los nuevos datos para el próximo mayo. De esta forma el período manejado por el programa empieza siempre con el mes en curso y cubre un período de doce meses.

Comentario

Línea 3615. Y es el número del mes que termina el período de doce meses.

Líneas 363∅-366∅. Esto es un ejemplo de cómo se llama al módulo 1.

Líneas 367∅-373∅. El nombre del mes se compara con el mes en curso del programa. Si no son iguales, el módulo comprueba el número del mes y si no, da un mensaje de error.

Líneas 374∅-4∅2∅. Esta sección del programa no puede comprobarse hasta que no se hayan entrado algunos datos, pero su propósito general es aceptar datos para actualizar la información si el mes ha cambiado desde su última utilización.

Línea 375∅. MO es el número del último mes en que fue utilizado el programa. M2 es el número del mes actual.

Línea 379∅. N1 es el número total de títulos de pagos que han sido entrados previamente.

Líneas 38∅∅-386∅. Sucesivamente se pide una nueva cifra para cada título de pago. Como guía, se muestra la cifra para el mes que debe ser borrado.

Líneas 39∅∅-397∅. El proceso se repite para los ingresos principales y adicionales.

Línea 4Ø1Ø. La subrutina llamada por esta línea recalcula el análisis presupuestario.

Comprobación del módulo 2.1.3

Tan sólo podemos comprobar si el módulo es capaz de tratar los cambios de mes y reconocer nombres inválidos de mes. Para hacerlo, una vez entrado MO en modo directo, empiece el programa con un GOTO 1 e intente entrar algunos nombres inválidos de mes. Deberán ser rechazados. Si se entra el mes correcto, se producirá el retorno desde el módulo; tendrá que insertar temporalmente una orden STOP en la línea 3589 para evitar que se vuelva a reejecutar el módulo. Intente también entrar el mes que va después del mes actual; el programa deberá detenerse con un mensaje de error 2, en la línea 379Ø.

MODULO 2.1.4

```
1510>REM ************************
1520 REM ESTABLECIMIENTO DE
         PAGOS REGULARES
1530 REM ************************
1540 DIM B$(2,20,9)
1550 DIM C(20)
1560 DIM B(2,20,12)
1570 DIM D(4,12)
1580 DIM E(4,12)
1590 DIM F(2,12)
1620 LET N1=0
1630 LET N2=0
1635 DIM T(2)
1640 GO SUB 4030
1650 GO SUB 3050
1660 GO SUB 2900
1670 RETURN
```

Este módulo declara las distintas tablas utilizadas para guardar los datos y pide las entradas iniciales para las mismas. Obsérvese que la llamada de este módulo da como resultado la pérdida de cualquiera de los datos allí guardados.

Comentario

Línea 154Ø. B$ es la tabla donde guardaremos los nombres de los títulos de los pagos: está prevista para 2Ø títulos. La primera dimensión de la tabla es 2, ya que el programa permite dos conjuntos paralelos de datos, uno de ellos son las cifras reales y el otro una copia en la que puede realizar cambios hipotéticos sin que afecten a los datos reales.

Línea 155Ø. C es la tabla que contendrá, para cada título de pago, el pago mensual promedio necesario para cubrir el gasto anual.

Línea 156Ø. La tabla B contendrá los pagos mensuales para los datos reales e hipotéticos.

Línea 157Ø. La tabla D contendrá los pagos mensuales totales para cada mes. Contendrá también una comprobación que indique si el presupuesto mensual promedio ha cubierto realmente la cantidad que debía pagarse en este mes.

Línea 158Ø. La tabla E contiene detalles mensuales de los ingresos reales e hipotéticos.

Línea 159Ø. La tabla F contiene el balance entre ingresos y gastos.

Líneas 162Ø-163Ø. N1 y N2 se refieren al número de títulos de pagos registrados en los lados real e hipotético de los datos.

Líneas 164Ø-166Ø. Una vez inicializadas las tablas se pide al usuario que realice una entrada inicial de ingresos y pagos. Entonces estos datos serán copiados en el lado hipotético de la tabla por la subrutina de la línea 29ØØ.

Comprobación del módulo 2.1.4

La comprobación de este módulo debe expresar hasta la utilización de las tablas por otros módulos.

MODULO 2.1.5

```
1190>GO SUB 3590
1210 PAPER 7: INK 0: CLS : PRINT
 "PRESUPUESTO DOMESTICO"
1220 PRINT '"FUNCIONES DISPONIBL
ES:"
1230 PRINT PAPER 5;"1)INICIALIZA
R"
1240 PRINT "2)BORRAR PRESUPUESTO
 HIPOTETICO"
1250 PRINT PAPER 5;"3)MOSTRAR AN
ALISIS MENSUAL"
1260 PRINT "4)ANALISIS HIPOTETIC
O"
1270 PRINT PAPER 5;"5)CAMBIOS RE
ALES"
1280 PRINT "6)CAMBIOS HIPOTETICO
S"
1290 PRINT PAPER 5;"7)TITULOS NU
EVOS"
1300 PRINT "8)TITULOS HIPOTETICO
S"
1302 PRINT PAPER 5;"9)ELIMINAR T
ITULO REAL"
1304 PRINT "10)ELIMINAR TITULO H
IPOTETICO"
1310 PRINT PAPER 5;"11)FINALIZAR
"
```

```
1330 LET P1=15
1340 LET P$="CUAL PRECISA?"
1350 GO SUB 3360
1360 CLS
1370 LET Z$=Q$
1380 LET H=1+(Z$="2")+(Z$="4")+(
Z$="6")+(Z$="8")+(Z$="10")
1390 IF Z$="1" THEN GO SUB 1520
1400 LET N=N1
1410 IF H=2 THEN LET N=N2
1420 IF Z$="2" THEN GO SUB 2900
1430 IF Z$="3" OR Z$="4" THEN GO
 SUB 1690
1440 IF Z$="5" OR Z$="6" THEN GO
 SUB 4290
1450 IF Z$="7" OR Z$="8" THEN GO
 SUB 3050
1452 IF Z$="9" OR Z$="10" THEN G
O SUB 4870
1460 IF Z$="11" THEN GO TO 1490
1470 CLS
1480 GO TO 1200
1490 PRINT FLASH 1;AT 10,0;"DESE
A REGRABAR? (S/N)": INPUT Q$: CL
S : IF Q$="S" THEN SAVE "PRESUP"
: BEEP 1,2: PRINT AT 10,0;"REBOB
INAR Y LUEGO PULSAR UNA"'"TECLA
PARA VERIFICAR": PAUSE 0: VERIFY
 "PRESUP": STOP
1500 STOP
```

Este es un módulo estándar de menú.

Comentario

Línea 138∅. No hay secciones separadas del programa para tratar con la manipulación de los datos hipotéticos. Cada módulo del programa trabaja con los datos reales si H=1 y con los datos hipotéticos si H=2. La línea utiliza condiciones lógicas como variables.

Línea 139∅. Ya que no hay ningún motivo para que el número de títulos de pago sea el mismo en las secciones real e hipotética de los datos, la variable N se pone igual a N1 o N2 en función de si H es igual a 1 o es igual a 2.

Comprobación del módulo 2.1.5

Entre la línea 332∅ y pulse RETURN. Ahora el programa deberá aceptar entradas para este módulo con excepción de la función 1. Naturalmente no sucederá nada, excepto que la función STOP debe funcionar.

```
3040>REM ***********************
3050 REM ENTRADA DE PAGOS
3060 REM ***********************
3070 PRINT "      ENTRADA DE FAC
TURAS"
3080 PRINT ''"PRECEDER EL NOMBRE
 DEL ELEMENTO CON UN * SI NO SE
DESEA INCLUIR EN EL PRESUPUESTO.
"
3090 LET P$="TITULO (""ZZZ"" ACA
BAR):"
3100 LET P1=6
3110 GO SUB 3360
3115 IF Q$="ZZZ" THEN GO SUB 259
0: RETURN
3130 PRINT AT 6,0;P$;Q$
3140 IF H=1 THEN LET N1=N1+1
3150 IF H=2 THEN LET N2=N2+1
3160 LET N=N1*(H=1)+N2*(H=2)
3170 IF N<=20 THEN GO TO 3210
3180 PRINT AT 8,0;"ARCHIVO DE PA
GOS LLENO."
3190 PRINT ''"Pulsar cualquier t
ecla para"'"continuar."
3195 PAUSE 0
3200 RETURN
3210 LET B$(H,N)=Q$
3220 FOR I=MO TO Y
3230 LET I1=I-12*(I>12)
3240 LET P$="CANTIDAD PARA "+N$(
I1)+":"
3250 LET P1=I+7-MO
3260 GO SUB 3360
3270 LET B(H,N,I1)=VAL Q$
3280 PRINT AT I+7-MO,25;Q$
3290 NEXT I
3300 CLS
3320 GO TO 3070
```

El propósito de este módulo es aceptar nuevos títulos de pagos y pedir detalles de los pagos correspondientes a estos títulos para un período de doce meses.

Comentario

Línea 3Ø8Ø. Los elementos precedidos por un asterisco no se incluirán en el cálculo del presupuesto mensual promedio.

Línea 316Ø. Obsérvese cómo se utilizan las condiciones lógicas para incrementar N1 o N2.

Línea 321Ø. Obsérvese cómo se utiliza el valor de H para determinar en qué lado de la tabla deberán colocarse los datos.

Línea 323Ø. Si se ha pasado por el final del año, se restará doce de I1 para que permanezca dentro del rango del 1 al 12.

Línea 325Ø. El contador del bucle se utiliza para el formato de visualización de los sucesivos P$.

Comprobación del módulo 2.1.6

El programa, una vez inicializado, deberá aceptar títulos de pagos junto con los pagos asociados y guardarlos, ya sea en el lado real o en el hipotético de las tablas correspondientes. Esto tan sólo puede comprobarse visualizando elementos de las tablas B$ y B en modo directo.

MODULO 2.1.7

```
4030>REM ************************
4040 REM INGRESOS
4050 REM ************************
4060 PRINT "              INGRESOS
"
4070 PRINT ''"ENTRE POR FAVOR EL
 SALARIO DE    12 MESES:"
4080 FOR I=MO TO Y
4090 LET I1=I-12*(I>12)
4100 LET P$=N$(I1)+":"
4110 LET P1=I+5-MO
4120 GO SUB 3340
4130 LET E(H,I1)=VAL Q$
4140 PRINT AT I+5-MO,10;E(H,I1)
4150 NEXT I
4160 CLS
4170 PRINT "ENTRE A CONTINUACION
 CUALQUIER OTRO INGRESO PREVISTO
:"
4180 FOR I=MO TO Y
4190 LET I1=I-12*(I>12)
4200 LET P1=I+5-MO
4210 LET P$=N$(I1)+":"
4220 GO SUB 3360
4230 LET E(H+2,I1)=VAL Q$
4240 PRINT AT I+5-MO,10;E(H+2,I1
)
4250 NEXT I
4260 CLS
4270 RETURN
```

Este módulo acepta los datos de los ingresos para los doce meses, bajo los títulos de salario y otros ingresos anticipados.

Comentario

Líneas 4Ø8Ø-415Ø. Este bucle acepta las cifras del salario que deben colocarse en la tabla E.

Líneas 417Ø-425Ø. Lo mismo, pero para los ingresos suplementarios.

Comprobación del módulo 2.1.7

A este módulo se accede a través del módulo de inicialización, que deberá ser llamado desde el menú. Se le pedirá que entre títulos de pagos y después, cuando salga con un ZZZ, que entre los ingresos correspondientes a los dos títulos. El programa deberá volver al menú una vez hayan sido entradas las cifras de los ingresos.

MODULO 2.1.8

```
3500 >REM ************************
3510  REM DINERO$
3520  REM ************************
3530  LET M=M+1000.0001*(M>=0)-10
00.0001*(M<0)
3540  LET M$=(STR$ ABS M)(2 TO 4)
3550  IF ABS M>=2000 THEN LET M$=
"***"
3560  IF M<0 THEN LET M$=""+M$+""
3570  RETURN
```

Este módulo construye textos partiendo de los datos guardados que representan sumas de dinero. Esto se hace para que las cifras sean más fáciles de formatear.

Comentario

Línea 353Ø. Se suma 1ØØØ.ØØØ1 para asegurar que habrán tres cifras antes del punto decimal y dos después del mismo, sin alterar el valor de cualquier cifra que pudiera estar ya en estas posiciones. M es una variable temporal y el proceso no afecta al valor del número original.

Línea 355Ø. Al presentar los datos, el programa tan sólo puede visualizar los tres dígitos que van delante del punto decimal. Si esto es un inconveniente serio para usted, es muy fácil modificarlo para que pueda visualizar valores hasta 9999. Esta limitación no afecta a lo que está guardado, sólo a lo que se visualiza.

Línea 356Ø. El programa necesita poder visualizar números negativos y dejar bien claro que son negativos. Esto se hace visualizándolos con inversión de video. Esto no es aparente en el listado, pero el texto creado al final de la línea consiste en:

"[4 con CAPS SHIFT]"+M$+"[3 con CAPS SHIFT]"

Los dos textos aparentemente vacíos que se colocan a ambos lados de M$ añaden dos caracteres de control. El primero coloca todo lo que le sigue con inversión de video. El otro vuelve al video normal.

Comprobación del módulo 2.1.8

En modo directo, entrar LET M=1 y después GOSUB 35ØØ. Después visualice el M$ que ha sido creado. Pruebe esto con otros valores, incluyendo algunos números negativos.

MODULO 2.1.9

```
2590 REM ************************
2600 REM ACTUALIZAR PRESUPUESTO
2610 REM ************************
2620 LET T(H)=0
2630 FOR I=1 TO N
2640 LET PRESUP=0
2650 IF B$(H,I,1)="*" THEN GO TO
 2710
2660 FOR J=1 TO 12
2670 LET PRESUP=PRESUP+B(H,I,J)
2680 NEXT J
2690 LET C(I)=PRESUP/12
2700 LET T(H)=T(H)+C(I)
2710 NEXT I
2720 LET TTOTAL=0
2730 LET CUM=0
2740 FOR I=MO TO Y
2750 LET I1=I-12*(I>12)
2760 LET D(H,I1)=0
2770 FOR J=1 TO N
2780 LET D(H,I1)=D(H,I1)+B(H,J,I
1)
2790 NEXT J
2800 LET TTOTAL=TTOTAL+D(H,I1)
2810 FOR J=1 TO N
2820 IF B$(H,J,1)="*" THEN LET T
TOTAL=TTOTAL-B(H,J,I1)
2830 NEXT J
2840 LET D(H+2,I1)=T(H)*(I-MO+1)
-TTOTAL
2850 LET CUM=CUM+E(H,I1)+E(H+2,I
1)-D(H,I1)
2860 LET F(H,I1)=CUM
2870 NEXT I
2880 RETURN
```

Este módulo realiza los cálculos que proporcionan los distintos análisis de ingresos y gastos. De hecho, es mucho menos complejo de lo que parece ya que todo lo que se hace son sencillas sumas, restas y divisiones por doce.

Comentario

Línea 262Ø. T es una tabla de dos elementos que se utiliza para guardar la suma de los pagos individuales mensuales presupuestados para obtener un presupuesto mensual total.

Líneas 263Ø-271Ø. En este bucle se encuentran los pagos anua-

les totales para cada título de pagos y se dividen por doce para obtener un promedio mensual. La tabla C guarda el promedio mensual para cada título de pagos. Obsérvese que los nombres de pagos que empiezan con un asterisco están exentos de este proceso.

Líneas 274∅-279∅. La tabla D se carga con los totales de los pagos para cada mes.

Líneas 28∅∅-284∅. El total de los pagos de cada mes se guarda en la tabla D. Estas sumas se acumulan en TTOTAL y de TTOTAL se resta cualquier pago que el usuario no quiera incluir en el presupuesto promedio. A cada paso a través del bucle que empieza en 274∅, en TTOTAL se coloca el total de los pagos hasta el mes I1 inclusive, que pertenecen a títulos de pagos incluidos en el presupuesto mensual promedio. En la línea 284∅ se resta TTOTAL de I1 veces el presupuesto mensual promedio. La cifra resultante es el balance del presupuesto mensual hasta este mes con respecto al total de los pagos de los elementos presupuestados. Esto se guarda en la segunda mitad de la tabla D. Y si todo esto le suena tan confuso como me suena a mí, tenga fe y espere hasta tener la oportunidad de visualizar unas cuantas cifras sencillas y después léalo de nuevo.

Línea 285∅. El balance entre ingresos y gastos se guarda en CUM y estos valores se irán guardando como elementos sucesivos de la tabla F.

Comprobación del módulo 2.1.9

Este módulo será comprobado después de entrar el módulo siguiente.

MODULO 2.1.10

```
1680>REM ************************
1690 REM VISUALIZACION DE
         FACTURAS
1700 REM ************************
1710 PRINT "               PAGOS"
1720 LET P$="MES DE INICIO"
1730 LET P1=3
1740 GO SUB 3360
1750 FOR I=1 TO 12
1760 IF T$=N$(I) THEN GO TO 1790
1770 NEXT I
1780 GO TO 1720
1790 LET M1=I
1800 IF MO-M1+12*(M1>MO-1)<4 THE
N LET M1=MO-4+12*(MO<5)
1810 CLS
1820 PRINT "MES         ■   ■   ■
 ■   ■ P ■"
1830 OVER 1: PRINT AT 0,0;TAB 10
;N$(M1, TO 3);TAB 14;N$(M1+1-12*
(M1>11), TO 3);TAB 18;N$(M1+2-12
```

```
*(M1>10) , TO 3);TAB 22;N$(M1+3-1
2*(M1>9) , TO 3)'L$: OVER 0
1840 FOR I=1 TO N
1843 PAPER 5: IF I/2<>INT (I/2)
THEN PAPER 7
1850 IF I<>11 THEN GO TO 1920
1860 PRINT AT 20,0;"PULSAR CUALQ
UIER TECLA PARA      BORRAR PANTA
LLA Y CONTINUAR"
1870 PAUSE 0
1880 FOR J=2 TO 21
1890 PRINT AT J,0;O$
1900 NEXT J
1910 PRINT AT 2,0;
1920 PRINT B$(H,I);"█";
1930 FOR J=M1 TO M1+3
1940>LET M=B(H,I,J-12*(J>12))
1950 GO SUB 3500
1960 PRINT M$;"█";
1970 NEXT J
1980 LET M=C(I)
1990 GO SUB 3500
2000 PRINT M$;"█"
2010 GO SUB 3500
2020 NEXT I
2030 PRINT K$
2040 PRINT ''"PULSAR CUALQUIER T
ECLA PARA      ANALISIS"
2050 PAUSE 0
2070 FOR I=2 TO 21
2080 PRINT AT I,0;O$
2090 NEXT I
2110 PAPER 5: PRINT AT 2,0;"TOTA
L    █"''"BAL.PRES.█"''"OTROS IN
G█"''"BAL. EFE.█"
2115 PAPER 7: PRINT AT 3,0;"PRES
UP   █"''"PAGA      █"''"INGR. TO
T█"''"BAL. ACU.█"
2130 FOR I=M1 TO M1+3
2140 OVER 1: PRINT AT 2,0;
2150 LET I1=I-12*(I>12)
2160 LET I2=10+4*(I-M1)
2170 LET M=D(H,I1)
2180 GO SUB 3500
2190 PAPER 5: PRINT TAB I2;M$;"█
"
2200 LET M=T(H)
2210 GO SUB 3500
2220 PAPER 7: PRINT TAB I2;M$;"█
"
2230 LET M=D(H+2,I1)
2240 GO SUB 3500
2280 PAPER 5: PRINT TAB I2;M$;"█
"
2290 LET M=E(H,I1)
2300 GO SUB 3500
2310 PAPER 7: PRINT TAB I2;M$;"█
"
2320 LET M=E(H+2,I1)
2330 GO SUB 3500
2340 PAPER 5: PRINT TAB I2;M$;"█
"
2350 LET M=E(H,I1)+E(H+2,I1)
2360 GO SUB 3500
2370 PAPER 7: PRINT TAB I2;M$;"█
"
2380 LET M=(E(H,I1)+E(H+2,I1)-D(
H,I1))
2400 GO SUB 3500
2440 PAPER 5: PRINT TAB I2;M$;"█
"
```

```
2450 LET M=F(H,I1)
2460 GO SUB 3500
2500 PAPER 7: PRINT TAB I2;M$;"■
"
2510 NEXT I
2520 OVER 0: PRINT K$( TO 28)
2530 LET P$="DESEA VER DE NUEVO
LAS CIFRAS?  (S/N):"
2540 LET P1=20
2550 GO SUB 3360
2560 IF Q$<>"S" THEN RETURN
2570 CLS
2580 GO TO 1810
```

El propósito de este módulo, más bien impresionante, es visualizar los distintos datos y resultados en forma tabular. De particular interés es la forma en que la tabla se formatea basándose principalmente en variables que ya han aparecido en el programa; si los datos tienen una estructura regular pueden utilizarse para dictar el formato de su presentación sin demasiadas dificultades.

Comentario

Líneas 17ø0-181ø. Esta sección acepta la entrada de un mes inicial para la tabla y comprueba su validez. El mes inicial no puede ser posterior al mes número 9 del período de doce meses ya que se visualizan cuatro meses.

Líneas 182ø-183ø. Estas dos líneas visualizan el encabezamiento de la tabla, incluyendo las tres primeras letras del nombre de cada mes.

Líneas 184ø-2ø2ø. Este bucle visualiza los títulos de los pagos y los pagos mensuales asociados con ellos para el período de cuatro meses. Para hacer resaltar las líneas separadas de la tabla, se varía el color del «papel» en función de si la variable del bucle es par o impar.

Líneas 2ø6ø-2ø9ø. Utilizar una línea de espacios vacíos y un bucle es una forma efectiva para borrar una parte de la pantalla.

Líneas 211ø-251ø. Esta sección visualiza los distintos análisis de ingresos y gastos. Obsérvese cómo se utiliza el contador del bucle (215ø, 216ø) para crear dos variables más en las que se basan el formateado de los datos. Tras la visualización de los títulos en las líneas 211ø-212ø, el examen de las líneas siguientes muestra que se componen de ocho secciones. Cada sección empieza con la formación de la variable temporal M, y la llamada del módulo 8, y a continuación la visualización de M$.

Comprobación del módulo 2.1.10

Borre cualquier dato guardado previamente, entre de nuevo MO en modo directo y empiece el programa con GOTO 1. Entre los ingresos y algunos pagos, pero que sean valores sencillos —decenas o centenas—. Llame al módulo 3 y examine el resultado. Si aparece algún problema con el formato es que probablemente habrá cometido algún error al entrar el módulo 10. Si las cifras no tienen sentido, el error está probablemente en el módulo 9. Si los pagos son incorrectos busque en la entrada de pagos en el módulo 6. Si los errores son en las cifras de ingresos, inspeccione el módulo 7.

MODULO 2.1.11

```
2890 >REM  ************************
2900  REM  PREPARAR TABLA
           HIPOTETICA
2910  REM  ************************
2920  FOR  I=1 TO N1
2930  LET  B$(2,I)=B$(1,I)
2940  FOR  J=1 TO 12
2950  LET  B(2,I,J)=B(1,I,J)
2960  LET  D(2,J)=D(1,J)
2970  LET  D(4,J)=D(3,J)
2980  LET  E(2,J)=E(1,J)
2990  LET  E(4,J)=E(3,J)
2995  LET  F(2,J)=F(1,J)
3000  NEXT J
3010  NEXT I
3020  LET  N2=N1
3030  RETURN
```

Este módulo copia los datos situados en la mitad real de la tabla y los pasa a la mitad hipotética. Si existía alguna diferencia entre las mitades real e hipotética, entonces se perderá cuando se utilice esta función.

Comprobación del módulo 2.1.11

Ahora ya está en situación de comprobar la existencia de una separación adecuada entre las áreas de datos reales e hipotéticos. Entre algunos pagos hipotéticos nuevos, compruebe que han sido aceptados llamando a la función 4 del menú, y después vuelva al menú y llame a la función 3. La tabla visualizada no debe mostrar ninguna traza de los pagos hipotéticos que ha entrado.

```
4280>REM ***********************
4290 REM CAMBIOS
4300 REM ***********************
4310 PRINT "               CAMBIOS"
4320 PRINT ''"1)CAMBIAR TITULO E
XISTENTE"'"2)CAMBIAR INGRESO PRI
NCIPAL"'"3)CAMBIAR INGRESO ADICI
ONAL"
4340 LET P$="CUAL DESEA?"
4350 LET P1=7
4360 GO SUB 3360
4370 LET H$=Q$
4380 CLS
4410 IF H$="1" THEN GO SUB 4460
4420 IF H$="2" OR H$="3" THEN GO
 SUB 4690
4440 GO SUB 2600
4450 RETURN
4460 LET P$="NOMBRE DEL TITULO A
 CAMBIAR?"
4470 LET P1=1
4480 GO SUB 3340
4490 FOR I=1 TO N
4500 IF T$=B$(H,I) THEN GO TO 45
60
4510 NEXT I
4520 PRINT ''" NO HAY TITULO CON
 ESTE NOMBRE."''"  Pulsar cualqu
ier tecla para     continuar."
4530 PAUSE 0
4550 RETURN
4560 LET B=I
4570 LET P$="NUEVA CIFRA O ""Z""
 PARA SALIR."
4580 LET P1=17
4590 FOR I=MO TO Y
4600 LET I1=I-12*(I>12)
4610 PRINT AT I-MO+3,0;N$(I1);":
";B(H,B,I1)
4620 GO SUB 3340
4630 IF Q$="Z" THEN GO TO 4660
4640 LET B(H,B,I1)=VAL Q$
4650 PRINT AT I-MO+3,15;"(";B(H,
B,I1);")"
4660 NEXT I
4670 GO SUB 2590
4680 RETURN
4730 LET X=2*(H$="3")
4750 PRINT "INGRESO PRINCIPAL:"
AND (H$="2");"INGRESO SUPLEMENTA
RIO:" AND (H$="3")
4760 LET P$="ENTRAR NUEVA CIFRA
O ""Z"" PARA    SALIR"
4770 LET P1=17
4780 FOR I=MO TO Y
4790 LET I1=I-12*(I>12)
4800 PRINT AT I+3-MO,0;N$(I1);":
";E(H+X,I1)
4810 GO SUB 3340
4820 IF Q$="Z" THEN GO TO 4850
4830 LET E(H+X,I1)=VAL Q$
4840 PRINT AT I-MO+3,15;"(";E(H+
X,I1);")"
4850 NEXT I
4860 RETURN
```

Este módulo permite realizar cambios sobre los datos ya existentes.

Comentario

Líneas 446∅-468∅. Se visualizan los pagos mensuales en curso, bajo los títulos de pagos especificados y estos pueden confirmarse o modificarse. A continuación se recalcula el análisis presupuestario.

Líneas 473∅-486∅. La variable X se coloca en función de si se ha especificado un ingreso principal o adicional. Las cifras entradas se guardan en la sección correspondiente en la tabla E, en función del valor de X.

Comprobación del módulo 2.1.12

Llame a este módulo y cambie alguno de los datos ya existentes, tanto en el lado real como en el hipotético.

MODULO 2.1.13

```
4870 REM ***********************
4880 REM ELIMINAR TITULO
4890 REM ***********************
4900 LET P$="NOMBRE DEL TITULO A
 ELIMINAR?"
4910 LET P1=1
4920 GO SUB 3340
4930 FOR I=1 TO N
4940 IF T$=B$(H,I) THEN GO TO 50
00
4950 NEXT I
4960 PRINT AT 2,4;"TITULO NO ENC
ONTRADO."''" Pulsar cualquier te
cla para      continuar."
4970 PAUSE 0
4980 RETURN
5000 IF H=1 THEN LET N1=N-1
5010 IF H=2 THEN LET N2=N-1
5015 LET N=N1*(H=1)+N2*(H=2)
5020 FOR J=I TO N
5030 LET B$(H,J)=B$(H,J+1)
5040 FOR K=1 TO 12
5050 LET B(H,J,K)=B(H,J+1,K)
5060 NEXT K
5070 NEXT J
5120 GO SUB 2600
5130 RETURN
```

Este módulo borra un título de pago existente, tanto en el lado real como en el hipotético, junto con sus pagos asociados.

Comentario

Líneas 493∅-498∅. En nombre del elemento que hay que borrar se compara con los títulos de pagos existentes.

Líneas 5ØØØ-512Ø. Se decrementa N1 o N2 en función de si lo borrado corresponde a un dato real o hipotético. Después los elementos de B$ y B se desplazan hacia abajo, uno a uno, para superponerse a la posición del dato que hay que borrar. Se vuelve a calcular el análisis presupuestario.

Comprobación del módulo 2.1.13

La comprobación se consigue borrando sencillamente algunos títulos. Si el borrado funciona, el programa estará entrado correctamente y listo para su utilización.

Resumen

Este largo programa es una herramienta muy potente si se utiliza de forma adecuada, aunque se necesita cierta práctica para conseguir el máximo rendimiento. Si se toma seriamente puede conseguir alguna información sorprendente sobre el estado de sus finanzas a lo largo del año — cuándo habrá que apretarse el cinturón o cuándo quedará algo para malgastar, cómo podrían reorganizarse los pagos para estar seguro de que queda un poco más para Navidad o para las vacaciones, cuál sería el efecto total de una nueva obligación o del aumento de los ingresos.

Sin embargo, esto no es todo. Sin demasiadas modificaciones, la estructura del programa podría aplicarse a una gran variedad de aplicaciones donde los datos deban visualizarse y analizarse mediante una serie de títulos, donde necesiten cambiarse a voluntad y donde puedan realizar entradas o cambios hipotéticos sin modificar los datos ya existentes.

Con cambios en el módulo 9 puede conseguirse que el programa realice cálculos muy distintos de los que se han especificado aquí. Recuerde que lo más difícil no es conseguir que funcionen las líneas individuales de BASIC cuando se emprende una nueva aplicación; es el conseguir que funcionen todas juntas. Una vez que este programa funcione satisfactoriamente, con las mejoras que desee realizar, considérelo como una estructura para sus propias ideas y tareas, en lugar de considerarlo como algo que tan sólo puede aplicarse a su presupuesto familiar.

Además, tiene que haber obtenido cierta confianza, al entrar este programa, en la técnica de manejar y subdividir grandes tablas mediante la simple utilización de variables dentro del programa. Contiene ejemplos prácticos sobre la forma en que pueden utilizarse las variables de los bucles y otras para ayudar al formateo de los datos.

Y finalmente, contiene un ejemplo de un sencillo módulo para formatear y comprobar los mensajes del programa, algo muy interesante para los programas interactivos.

Posibles mejoras

1) Al igual que Archivo, este programa está escrito utilizando muy poco las líneas multisentencia. Acorte el programa tanto como pueda combinando líneas individuales.
2) Excepto donde era absolutamente necesario para la claridad de la presentación, no se ha hecho ningún intento para conseguir que la visualización de salida del programa fuese interactiva en cuanto a los colores. Recorra el programa e ilumínelo un poco con algunas órdenes de color colocadas en los lugares correctos.
3) Eche un vistazo a las tablas definidas en el módulo 4. Varias de ellas podrían fácilmente combinarse entre sí. Esto podría dar como resultado un gran ahorro a la hora de visualizar la tabla de los análisis ya que el módulo podría trabajar metódicamente sobre una tabla en lugar que tener que referirse a 4 o 5. Pruébelo y véalo.

2.2 Contabilidad

Este sencillo programa permite al usuario crear unas cuentas claras y comprensibles. No se realiza ningún análisis excepto para generar totales y subtotales donde sea apropiado, por lo tanto si los libros no cuadran tendrá que descubrirlo usted mismo. Ya que el programa es bastante sencillo, los comentarios en los módulos son mucho más cortos que antes.

MODULO 2.2.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 PAPER 7: INK 0: CLS : PRINT
"          CONTABILIDAD"
1040 PRINT '"FUNCIONES DISPONIBL
ES:"
1050 PRINT '"1)ENTRAR NUEVOS TIT
ULOS"
1060 PRINT '"2)CAMBIAR IMPORTES/
BORRAR ELEM."
1070 PRINT '"3)PRESENTAR CUENTAS
"
```

```
1080 PRINT '"4)INICIALIZAR CUENT
AS"
1090 PRINT '"5)FINALIZAR"
1100 INPUT Z$
1110 CLS
1120 IF Z$<>"4" AND Z$<>"5" THEN
 GO SUB 1320
1130 CLS
1140 IF Z$="1" THEN GO SUB 1390
1150 IF Z$="2" THEN GO SUB 2120
1160 IF Z$="3" THEN GO SUB 2660
1170 IF Z$="4" THEN GO SUB 1210
1180 IF Z$="5" THEN GO TO 1202
1190 CLS
1200 GO TO 1000
1204 PRINT FLASH 1;AT 10,6;"CONT
ABILIDAD": INPUT "Desea regrabar
? (S/N) ";Q$: IF Q$="S" THEN SAV
E "CONTAB": BEEP 1,2: PRINT AT 1
0,0;"REBOBINAR Y LUEGO PULSAR CU
ALQUIER TECLA PARA VERIFICAR": V
ERIFY "CONTAB"
1208 STOP
```

Este es un programa estándar de menú.

MODULO 2.2.2

```
1210>REM ***********************
1220 REM HABER/DEBE
1230 REM ***********************
1240 DIM A$(2,100,15)
1250 DIM A(2,100)
1260 DIM C(2)
1270 LET C(1)=1
1280 LET C(2)=1
1290 DIM O$(32)
1300 LET L$="-------"
1310 RETURN
```

Este módulo inicializa las variables utilizadas y producirá la pérdida de cualquier dato guardado previamente en ellas.

Comentario

Línea 124Ø. A$ contendrá los nombres de los títulos en dos secciones, una para el haber y la otra para el debe.

Línea 125Ø. La tabla contendrá las cifras correspondientes a los títulos guardados en A$.

Líneas 126Ø-128Ø. C contendrá el número de elementos guardados en las áreas de debe y haber.

MODULO 2.2.3

```
1320 >REM ************************
1330 REM HABER O DEBE?
1340 REM ************************
1350 PRINT AT 10,3;"SELECCIONAR:
1) HABER"
1360 PRINT "                2) DEB
E"
1370 INPUT CD
1380 RETURN
```

Este módulo coloca la variable CD en función de si el usuario desea tratar con el lado de los datos correspondientes al haber o al debe.

Comprobación del módulo 2.2.3

Cualquier función seleccionada desde el menú deberá dar como resultado la llamada de este módulo.

MODULO 2.2.4

```
1390 >REM ************************
1400 REM ENTRAR ELEMENTOS
1410 REM ************************
1430 PRINT "          NUEVOS ELEMEN
TOS"
1440 PRINT '"SELECCIONAR: 1)ELEME
NTO SIMPLE"
1450 PRINT "             2)TITULO
 PRINCIPAL"
1460 PRINT "             3)SUBTIT
ULO"
1470 PRINT '"           ""0"" PA
RA SALIR"
1480 INPUT TIPO
1490 IF TIPO=0 THEN RETURN
1500 IF TIPO=3 THEN GO TO 1750
```

Este módulo se utiliza para especificar el tipo de título que va a ser guardado en las cuentas. Existen tres tipos:

1) Elementos individuales que serán entrados directamente en la columna principal de las cuentas.
2) Títulos principales, que no tienen cifras asociadas, pero que actúan como indicadores de grupos de:
3) subtítulos, que quedarán más desplazados a la derecha y sobre los que se realizará el subtotal separadamente de la columna principal de cifras.

```
1510 REM ************************
1520 REM ELEMENTO SIMPLE O
         ELEMENTO PRINCIPAL
1530 REM ************************
1540 LET Q=0
1550 PRINT AT 8,0;"NOMBRE? ";
1560 INPUT Q$
1570 PRINT Q$
1580 IF TIPO=2 THEN GO TO 1620
1590 PRINT '"IMPORTE? ";
1600 INPUT Q
1610 PRINT Q
1620 INPUT "Es correcto? (S/N)";
R$
1640 IF R$="S" THEN GO TO 1690
1650 FOR I=8 TO 14
1660 PRINT AT I,0;O$
1670 NEXT I
1680 GO TO 1550
1690 LET Q$=" "+Q$
1700 IF TIPO=2 THEN LET Q$(1)="*
"
1710 LET A$(CD,C(CD))=Q$
1720 LET A(CD,C(CD))=Q
1730 LET C(CD)=C(CD)+1
1735 CLS
1740 GO TO 1410
```

Este módulo acepta la entrada de títulos principales o de elementos individuales, según se definieron anteriormente.

Comentario

Líneas 154ø-17øø. Un nombre de elemento se acepta junto con una cantidad si la variable TIPO está a 1 —lo que indica un elemento individual—. Si TIPO está a 2, entonces se añade un asterisco delante del título, como indicador de que se trata de un título principal.

Líneas 171ø-174ø. Si el elemento no es un título principal, entonces la variable Q contendrá la cantidad correspondiente al elemento. En esta sección el título está guardado en A$ y la cantidad en A. Dependiendo de si el elemento corresponde al haber o al debe, se incrementa uno de los valores de C.

Comprobación del módulo 2.2.5

Entre algunos títulos de ambos tipos y compruebe que se han guardado en A$, junto con la cantidad asociada en A. Naturalmente las tablas deberán haber sido inicializadas por el módulo de inicialización.

```
1750 REM ************************
1760 REM SUBTITULO
1770 REM ************************
1780 PRINT '"NOMBRE DEL TITULO P
RINCIPAL? "';
1790 INPUT Q$
1800 PRINT Q$
1810 LET Q$="*"+Q$
1812 DIM T$(15)
1814 LET T$=Q$
1820 FOR I=1 TO C(CD)
1830 IF A$(CD,I)=T$ THEN GO TO 1
890
1840 NEXT I
1850 PRINT '"PERDON, NO HAY UN T
ITULO CON ESTE NOMBRE."
1860 PRINT ''"Pulsar cualquier t
ecla para       seguir.": PAUSE 0
1880 RETURN
1890 LET LUGAR=I+1
1900 PRINT '"NOMBRE DEL SUBTITUL
O? ";
1910 INPUT Q$
1920 PRINT Q$
1930 PRINT '"IMPORTE? ";
1940 INPUT Q
1950 PRINT Q
1960 INPUT "Son correctos? (S/N)
 ";R$
1980 IF R$="S" THEN GO TO 2030
1990 FOR I=8 TO 16
2000 PRINT AT I,0;O$
2010 NEXT I
2015 PRINT AT 8,0;
2020 GO TO 1900
2030 LET Q$="£"+Q$
2040 FOR I=C(CD)+1 TO LUGAR+1 ST
EP -1
2050 LET A$(CD,I)=A$(CD,I-1)
2060 LET A(CD,I)=A(CD,I-1)
2070 NEXT I
2080 LET A$(CD,LUGAR)=Q$
2090 LET A(CD,LUGAR)=Q
2100 LET C(CD)=C(CD)+1
2105 CLS
2110 GO TO 1410
```

Sirve para aceptar subtítulos tal como ya se definió.

Comentario

Líneas 175∅-189∅. Se entra el nombre del título principal correspondiente, comparándolo con los títulos ya guardados y la posición de este título principal se guarda en la variable LUGAR.

Líneas 19∅∅-2∅3∅. El nombre del nuevo subtítulo y la cantidad asociada son entrados y confirmados por el usuario.

Líneas 2∅4∅-21∅∅. Se crea un espacio moviendo todos los elementos, desde la posición LUGAR en adelante, un espacio hacia

arriba en la tabla. El nuevo subtítulo es entonces colocado inmediatamente después del título principal correspondiente.

Comprobación del módulo 2.2.6

Entre un subtítulo que esté bajo uno de los títulos principales que entró al realizar la comprobación del último módulo. Compruebe que el título y la cantidad asociada han sido colocados en las tablas correspondientes.

MODULO 2.2.7

```
2980>REM ************************
2990 REM CREAR M$
3000 REM ************************
3005 LET M=M+.001
3010 LET M$=(STR$ M)( TO LEN ST
R$ M-1)
3080 RETURN
```

Este módulo genera un texto estandarizado, con dos cifras decimales, a partir de las cifras asignadas a los distintos títulos. A diferencia del módulo similar del programa anterior, no se añaden ceros al principio del número.

MODULO 2.2.8

```
2660>REM ************************
2670 REM PRESENTAR CUENTAS
2680 REM ************************
2690 PRINT AT 0,12;"HABER" AND C
D=1;AT 1,12;"DEBE" AND CD=2
2700 LET TTOTAL=0
2710 LET STOTAL=0
2720 FOR I=1 TO C(CD)-1
2730 IF I>1 AND A$(CD,I,1)<>"£"
THEN PRINT
2740 LET TTOTAL=TTOTAL+A(CD,I)
2750 IF A$(CD,I,1)="£" THEN PRIN
T TAB 2;
2760 PRINT A$(CD,I,2 TO );
2770 IF A$(CD,I,1)="*" THEN GO T
O 2850
2780 IF A(CD,I)=0 THEN GO TO 285
0
2790 LET M=A(CD,I)
2800 GO SUB 2980
2810 PRINT TAB 25-LEN M$;
2820 IF A$(CD,I,1)<>"£" THEN PRI
NT TAB 32-LEN M$;
2830 IF A$(CD,I,1)="£" THEN LET
STOTAL=STOTAL+A(CD,I)
2840 PRINT M$
```

```
2850 IF STOTAL=0 OR A$(CD,I+1,1)
="£" THEN GO TO 2910
2860 PRINT TAB 18;L$
2870 LET M=STOTAL
2880 GO SUB 2980
2890 PRINT TAB 25-LEN M$;M$
2900 LET STOTAL=0
2910 NEXT I
2920 LET M=TTOTAL
2930 GO SUB 2980
2950 PRINT '"TOTAL:";TAB 32-LEN
M$;M$
2960 PRINT '"Pulsar cualquier te
cla para      seguir."
2965 PAUSE 0
2970 RETURN
```

El propósito de este módulo es visualizar o bien el lado del haber o el del debe, de las cuentas.

Comentario

Línea 269∅. Obsérvese la utilización de AND para controlar lo que aquí se visualiza. Cuando dos elementos están conectados por AND, toman el valor del primer elemento del par si el segundo elemento tiene un valor positivo. Ya hemos visto que las condiciones como CD=2 tienen un valor de 1 o ∅ en función de si la condición es verdadera o falsa. De acuerdo con ello, en esta línea, si CD=1, entonces la expresión «HABER» AND CD=1 toma el valor «HABER». Si CD no es 1, entonces la expresión no tiene ningún valor y no se visualiza nada. Esta es una forma muy económica de hacer elecciones entre elementos que deben visualizarse. (Para más información sobre el extraño comportamiento de las expresiones lógicas, ver el capítulo 13 del manual de Sinclair.)

Líneas 27∅∅-271-. Estas dos variables se utilizarán para registrar los totales de los distintos grupos de subtítulos y el total de los totales.

Líneas 272∅-291∅. En este bucle se examina cada entrada de A$ para ver si se trata de un elemento individual, de un título principal o de un subtítulo. Si es un elemento individual, se visualiza el título y se entra la cantidad en la columna principal. Si se trata de un título principal entonces no se visualiza ninguna cantidad asociada. Si se trata de un subtítulo, entonces el título se desplaza dos espacios a la derecha y su cantidad asociada se visualiza en una columna separada, antes de la columna principal. Al final de cada grupo de subtítulos se visualiza el total del grupo al pie del mismo.

Comprobación del módulo 2.2.8

Entre algunos títulos y haga que se visualicen. Todos los puntos decimales de la columna principal deberán estar alineados.

MODULO 2.2.9

```
2120 >REM ***********************
2130 REM CAMBIOS Y ELIMINACIONES
2140 REM ***********************
2150 FOR I=1 TO C(CD)-1
2160 PRINT AT 0,8;"CAMBIAR O BOR
RAR"
2170 IF A$(CD,I,1)<>"£" THEN PRI
NT AT 3,0;O$
2180 PRINT AT 5,0;O$
2190 IF A$(CD,I,1)<>"£" THEN PRI
NT AT 3,0;A$(CD,I,2 TO )
2200 IF A$(CD,I,1)="£" THEN PRIN
T AT 5,0;A$(CD,I,2 TO )
2210 IF A(CD,I)=0 THEN GO TO 225
0
2220 LET M=A(CD,I)
2230 GO SUB 2980
2240 PRINT TAB 16;M$
2250 PRINT AT 18,0;">ENTER, SIGU
IENTE ELEMENTO"
2260 PRINT ">""CCC""=CAMBIAR IMP
ORTE"
2270 PRINT ">""ZZZ""=SALIR"
2280 PRINT ">""DDD""=BORRAR ELEM
ENTO"
2290 INPUT Q$
2300 IF Q$="ZZZ" THEN RETURN
2310 IF Q$="DDD" THEN GO SUB 246
0: RETURN
2325 IF Q$="" THEN GO TO 2440
2330 PRINT AT 18,0;O$;O$;O$;O$
2340 PRINT AT 18,0;"VARIACION DE
L IMPORTE? ";
2350 INPUT Q$
2360 PRINT Q$
2370 INPUT "Es correcto? (S/N)";
R$
2390 PRINT AT 18,0;O$;O$;O$;O$
2400 IF R$="N" THEN GO TO 2340
2410 LET A(CD,I)=A(CD,I)+VAL Q$
2420 PRINT AT 6,0;O$
2430 GO TO 2160
2440 NEXT I
2450 RETURN
```

Este módulo permite al usuario cambiar las cantidades asociadas a elementos que ya se han entrado en las cuentas. No existe una función de búsqueda asociada a este módulo; el usuario simplemente recorrerá los elementos de las cuentas hasta que llegue al que está buscando. La cantidad asociada a un elemento se cambia entrando una cantidad que se sumará a la cantidad actualmente guardada. Esto suponiendo que las modificaciones generalmente serán para registrar

gastos o ingresos extras bajo los títulos ya existentes. Si desea restar, tan sólo hay que entrar un número negativo.

Comprobación del módulo 2.2.9

Intente cambiar varios elementos y después visualizar sus nuevos valores.

MODULO 2.2.10

```
2460>REM ***********************
2470 REM ELIMINAR ELEMENTO
2480 REM ***********************
2490 LET LUGAR=I
2500 LET BORRADO=1
2510 IF A$(CD,LUGAR,1)<>"*" THEN
 GO TO 2550
2520 LET BORRADO=0
2530 LET BORRADO=BORRADO+1
2540 IF A$(CD,LUGAR+BORRADO,1)="
£" THEN GO TO 2530
2550 FOR K=LUGAR TO C(CD)-BORRAD
O-1
2560 LET A$(CD,K)=A$(CD,K+BORRAD
O)
2565 LET A(CD,K)=A(CD,K+BORRADO)
2570 NEXT K
2580 LET C(CD)=C(CD)-BORRADO
2650 RETURN
```

Este módulo borra elementos de las cuentas. Si el elemento que quiere borrarse es un título principal, también se borrarán todos los subtítulos asociados al mismo.

Comentario

Líneas 249∅-254∅. Si el elemento que hay que borrar es un título principal, la variable BORRADO se incrementa hasta que sea igual al título principal más el número de sus subtítulos.

Líneas 255∅-258∅. Los elementos que están por encima de la tabla se mueven hacia abajo para que cubran a los elementos que deben borrarse. La variable que registra el número de elementos será decrementada según el número de elementos borrados.

Comprobación del módulo 2.2.10

Borre algunos elementos y compruebe que han sido realmente borrados. El programa ya está ahora listo para su utilización.

Posibles mejoras

1) ¿Por qué no intentar añadir algunos interesantes colores más?
2) Intente añadir una función que realice un balance y lo incluya en uno u otro lado de las cuentas.
3) Si hubiera 5Ø elementos o más a un lado de las cuentas, se necesitaría demasiado tiempo para pasar a través de ellos para hacer modificaciones. Intente añadir una función de búsqueda que permita al usuario entrar un número y que entonces salte hacia adelante o incluso hacia atrás el número entrado de elementos.

2.3 Cuentas bancarias

En este programa empezaremos a hacer un uso extensivo de las líneas multisentencia. Este programa es una herramienta que permite mantener de una forma clara los registros financieros en forma parecida a una cuenta bancaria. Trata los pagos que deben ser realizados, tanto si son regulares o irregulares y los inserta en el registro mensual, en el día en que deben realizarse.

El programa es relativamente sencillo comparado con los que hemos hecho anteriormente, pero hay que señalar que no es tan sencillo como parece a primera vista, ya que en este programa, por vez primera, utilizaremos una considerable proporción de líneas multisentencia. Sin ellas el programa aparecería considerablemente más largo. Un aspecto a tener en cuenta al entrar este programa, o cualquier otro que utilice líneas multisentencia es el comportamiento de las sentencias IF. Estas sentencias son capaces de crear problemas si se utilizan inadecuadamente en las líneas multisentencia, creando errores en los programas que son extremadamente difíciles de detectar. De todas formas, las líneas multisentencia pueden utilizarse para aumentar la efectividad de las sentencias IF en virtud del hecho de que si la condición especificada por la sentencia IF no se cumple, el programa no sólo salta la parte de la línea directamente relacionada con la sentencia IF, sino que salta la totalidad del resto de la línea.

En otras palabras, cualquier sentencia colocada después de una sentencia IF, será ejecutada únicamente si la sentencia IF es cierta. Esto es tan distinto del comportamiento de las líneas monosentencia que es muy fácil cometer un error.

La ventaja de todo ello es que proporciona una forma de GOTO, automática y elegante, que ni tan sólo hay que especificar. Si se tiene

una serie de diez operaciones que deberán ser ejecutadas juntas siempre que, por ejemplo, C=1 en algún punto, entonces con una línea monosentencia se tendría que colocar una sentencia IF al principio de la sección que especificase un salto hasta después de las 1Ø operaciones si C no fuese igual a 1. Funciona, pero por otra parte es un poco confuso y hace que sea difícil leer un programa en el que haya muchos saltos de este tipo.

Sin embargo, con líneas multisentencia, podría empezarse una única línea con IF C=1 y a continuación las 1Ø operaciones. No sólo haría esto que funcionase, si no que ahorraría memoria y haría que el programa fuese más legible, ya que quedaría muy claro que las 1Ø operaciones forman una unidad lógica.

MODULO 2.3.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : INK 0: PAPER 7: PRINT
 AT 0,8; INK 2; FLASH 1;"CUENTAS
 BANCARIAS"
1040 PRINT '"1)NUEVOS PAGOS"
1050 PRINT '"2)EXAMINAR/BORRAR P
AGOS"
1060 PRINT '"3)IMPRIMIR INFORMES
"
1070 PRINT '"4)INICIALIZAR"
1080 PRINT '"5)FINALIZAR"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1250
1110 IF Z$="2" THEN GO SUB 1420
1120 IF Z$="3" THEN GO SUB 1550
1130 IF Z$="4" THEN GO SUB 1180
1140 IF Z$="5" THEN GO TO 1160
1150 CLS : GO TO 1000
1160 PRINT AT 10,8; FLASH 1; INK
 2;"CUENTAS BANCARIAS": INPUT "H
A ENTRADO ALGUN NUEVO DATO QUE D
ESEE GUARDAR? (S/N)";Q$: IF Q$="
S" THEN SAVE "CUENTAS": BEEP 1,2
: PRINT "REBOBINE Y A CONTINUACI
ON PULSE ""ENTER"" PARA VERIFICA
R": PAUSE 0: VERIFY "CUENTAS": P
RINT "     PROGRAMA VERIFICADO"
1170 STOP
```

Este es un módulo estándar de menú.

MODULO 2.3.2

```
1180>REM ************************
1190 REM VARIABLES
1200 REM ************************
1210 DIM A$(100,26): DIM A(100)
1220 DEF FN A$(X)=(STR$ (X+10000
```

```
.001))(2 TO 8)
1230 LET PAGOS=0
1240 RETURN
```

Comentario

Línea 121∅. A$ se utilizará para guardar los nombres y otra información referente a los pagos individuales. A contendrá las cantidades en sí.

Línea 122∅. Si ha seguido los dos programas anteriores no deberá tener ningún problema en identificar el propósito de esta función. En una única línea, esta función crea un formato estándar para cualquier cifra que se le dé, con un valor máximo de 9999.99. Debe recordar que en el módulo 2 de Archivo discutimos en qué casos una función definida por el usuario representaba un ahorro significativo con respecto a una subrutina de una o dos líneas. Esta función, con su único argumento, X, se utilizará para formatear dos variables distintas, ahorrándonos así la utilización de dos subrutinas cortas. Si quiere, cuando se haya familiarizado con su utilización en este programa, puede volver atrás hasta los dos programas anteriores y sustituir allí las dos subrutinas cortas utilizadas para estandarizar el formato de las cifras.

Línea 123∅. PAGOS se utiliza para registrar el número total de elementos en el archivo.

MODULO 2.3.3

```
1250>REM ************************
1260 REM NUEVO PAGO REGULAR
1270 REM ************************
1280 PRINT AT 0,10; FLASH 1; INK
 2;"PAGOS"
1290 PRINT '"1) HABER / 2)DEBE?
";: INPUT CD: PRINT CD
1300 PRINT '"NOMBRE DEL PAGO? ";
: INPUT Q$: PRINT Q$
1310 PRINT '"IMPORTE? ";: INPUT
Q: PRINT Q
1320 PRINT '"MESES:p.e. 01040710
 : ": INPUT R$: PRINT R$
1330 PRINT '"DIA DE PAGO? ";: IN
PUT S: PRINT S
1340 INPUT "SON CORRECTOS? (S/N)
";T$: IF T$="N" THEN CLS : GO TO
 1250
1350 LET PAGOS=PAGOS+1: FOR J=PA
GOS TO 2 STEP -1
1360 IF S<CODE A$(J-1,1) THEN LE
T A$(J)=A$(J-1): LET A(J)=A(J-1)
: NEXT J
1370 LET A$(J)="": LET A$(J,26)=
CHR$ CD: LET A$(J,14 TO 25)=Q$:
LET A(J)=Q
```

```
1380 IF CD=2 THEN LET A(J)=A(J)*
-1
1390: FOR I=1 TO LEN R$ STEP 2:
LET A$(J,1+VAL R$(I TO I+1))="1"
: NEXT I:
1400 LET A$(J,1)=CHR$ S
1410 RETURN
```

Este módulo acepta la entrada de nuevos elementos, incluyendo detalles como el nombre, cantidad, los meses en que se hace el pago y el día de pago. El módulo también toma nota de si el elemento corresponde al haber o al debe. Si corresponde al debe, la cantidad se guarda en forma negativa.

Comentario

Líneas 135∅-136∅. Empezando con el último elemento del archivo, el módulo recorre todos los elementos para buscar el primero de ellos que sea menor por orden alfabético que el elemento entrado. Esto clasificará a los elementos por orden del día de pago, ya que el día de pago se guarda en forma del código del carácter del primer carácter de cada línea de A$. Obsérvese que para insertar elementos individuales en una tabla, siempre es más eficiente empezar por el final de la lista de elementos, ya que entonces puede examinarse cada elemento y, si es necesario, desplazarlo un lugar hasta que se encuentre en la posición correcta. Esto elimina la necesidad de dos bucles separados, uno para buscar la posición correcta a través del archivo y otro para desplazar los elementos para crear espacio para la nueva entrada.

Líneas 137∅-14∅∅. La cantidad se guarda en A. El resto de la información se guarda toda en la misma línea de A$. El primer carácter se utiliza para guardar el día de pago: los doce siguientes se utilizan para guardar los meses en los que se realiza el pago. Las posiciones desde la 14 a la 25 se utilizan para el nombre del elemento. La posición 26 sirve para registrar que el elemento corresponde al haber o al debe.

Comprobación del módulo 2.3.3

Ejecute el programa y llame a la función de inicialización. Después entre algunos elementos y detenga el programa. Visualice el contenido de A$ y de A e intente relacionarlos con la descripción dada en el comentario de este módulo.

```
1550>REM ************************
1560 REM PREPARAR INFORME
1570 REM ************************
1580 LET TOTAL=0
1590 PRINT AT 0,10; INK 2; FLASH
 1;"INFORME"
1600 PRINT ''"NUMERO DE MES PARA
 EL INFORME:": INPUT Q: PRINT Q
1610 CLS : PRINT AT 0,10; INK 1;
"MES: ";Q
1620 FOR I=1 TO PAGOS
1630 IF A$(I,1+Q)<>"1" THEN GO T
O 1690
1640 IF A$(I,26)=CHR$ 1 THEN PAP
ER 6
1650 PRINT 'CODE A$(I,1);"*" AND
 A$(I,26)=CHR$ 1;TAB 3;":";A$(I,
14 TO 25);
1660 PRINT TAB 16;FN A$(ABS A(I)
);"█";: LET TOTAL=TOTAL+A(I): IF
 TOTAL<0 THEN INVERSE 1
1670 PRINT FN A$(ABS TOTAL): INV
ERSE 0
1680 PAPER 7
1690 NEXT I: INPUT """ENTER"" CO
NTINUAR";Q$
1700 RETURN
```

Este módulo acepta una entrada que especifique un mes, y después visualiza las cuentas correspondientes a todos los pagos y recibos de este mes.

Comentario

Línea 163Ø. Cualquier elemento que no tenga un 1 en la posición del carácter que corresponde al mes especificado es ignorado.

Línea 164Ø. El color de PAPER se coloca a amarillo para los elementos del haber.

Línea 165Ø. Obsérvese la utilización de AND para visualizar un asterisco junto a la fecha de los elementos del haber.

Líneas 166Ø-167Ø. FN A$ se aplica a ABS A(I) y a ABS TOTAL. Obsérvese la utilización que se hace aquí de ABS; la función daría un resultado sin sentido si se aplicase a un número negativo debido a la presencia del signo menos. Naturalmente, el ABS podría haberse incluido en la función.

Comprobación del módulo 2.3.4.

Visualice las cuentas para varios meses.

MODULO 2.3.5

```
1420>REM ************************
1430 REM BORRAR PAGOS
1440 REM ************************
1450 FOR I=1 TO PAGOS
1460 CLS : IF CODE A$(I,26)=1 TH
EN PRINT ' PAPER 6;"HABER"
1470 IF CODE A$(I,26)=2 THEN PRI
NT ' PAPER 6;"DEBE"
1480 PRINT "PAGO:";A$(I,14 TO 25
)''"IMPORTE:";FN A$(ABS A(I))
1490 PRINT '"MESES:";: FOR J=1 T
O 12: PRINT STR$ J AND A$(I,1+J)
="1";"/";: NEXT J
1500 PRINT ''"DIA DE PAGO:";CODE
 A$(I,1)
1510 PRINT PAPER 6;'"ORDENES:"'"
1) ""DDD"" BORRAR"'"2) ""ZZZ"" S
ALIR"'"3) ""ENTER"" PARA EL
                SIGUIENTE ELEMENTO.
         "
1520 INPUT Q$: IF Q$="DDD" THEN
FOR J=I TO PAGOS: LET A$(J)=A$(J
+1): LET A(J)=A(J+1): NEXT J: LE
T PAGOS=PAGOS-1: RETURN
1530 IF Q$="" THEN NEXT I: RETUR
N
1540 RETURN
```

Este módulo visualiza elementos del archivo y da al usuario la opción de borrarlos.

Comentario

Línea 152∅. Los elementos se desplazan hacia abajo en el archivo para cubrir al que debe borrarse. Obsérvese que esto significa que el elemento final de la tabla ahora estará duplicado ya que no se ha colocado nada encima de él. Esto no tiene importancia ya que PAGOS se ha decrementado en una unidad y lo que queda en la última posición del archivo queda invisible para el programa.

Comprobación del módulo 2.3.5

Si esta función funciona el programa ya está completo.

Resumen

Este es realmente un programa muy sencillo. Plantea la interesante cuestión de hasta dónde hay que llegar al aumentar la complejidad del programa para simplificar su utilización. Los programas so-

fisticados de cuentas bancarias suelen ofrecer la posibilidad de registrar cambios de año, frecuencias de pago, fechas límite, etc. En este programa hemos adoptado la sencilla solución de preguntar al usuario en qué mes debe realizarse el pago, eliminando así la necesidad de diferenciar entre los pagos que se hacen de una vez y la variedad de pagos a plazos distintos. Sospecho que a menos que sus transacciones financieras sean extremadamente complejas, no querrá meterse en problemas para escribir nuevas funciones del programa que traten los pagos mensuales sin que el usuario especifique los meses. Tenga cuidado con la sobrecomplejidad en sus propios programas. Si se encuentra con que ha dedicado diez o veinte líneas extra a funciones que el cerebro humano puede realizar igualmente rápido antes de entrar los datos, entonces está malgastando tiempo y esfuerzo.

Posibles mejoras

1) El programa no tiene la posibilidad de acumular un balance de un mes a otro. Intente añadirle uno.
2) El asterisco situado junto a la fecha de los elementos del haber en realidad está allí para hacer que estos elementos destaquen cuando se envíen a la impresora, la cual no registrará las diferencias del color del papel. Duplique la sentencia de PRINT en el módulo 4 con LPRINTs.
3) ¿Podría combinar la instrucción que sirve para invertir la visualización de un número negativo y colocarla dentro de la función FN A$?

3. Dibujar. Gráficos del Spectrum

En este capítulo examinaremos algunas de las posibilidades gráficas del Spectrum. Para agotar el tema de los gráficos del Spectrum sería necesario dedicarle un libro completo y aquí no es lo que intentaremos hacer. Nos concentraremos en gráficos sencillos que puedan utilizarse para aumentar la efectividad de varios programas. Dejaremos aparte áreas tan fascinantes como los gráficos tridimensionales, los gráficos con movimiento y las figuras creadas por funciones matemáticas. Esto no quiere decir que estas áreas no sean importantes, sino que sería imposible hacerles justicia.

En los programas que vienen a continuación examinaremos las posibilidades abiertas por la capacidad de gráficos definidos por el usuario e intentaremos resolver algunos de los problemas que aparecen en la creación y almacenaje de dibujos realizados sobre la totalidad de la pantalla.

Los programas presentados en este capítulo son:

1) Caracteres. Un programa diseñado para permitirle crear caracteres, definibles por usted mismo, para utilizarlos con otros programas.
2) Diccionario. Un método para guardar los distintos caracteres gráficos que haya definido y creado.
3) Tangram. Un programa que trata del dibujo de figuras utilizando la orden DRAW.
4) Artista. Un programa que le permitirá realizar dibujos sobre la totalidad de la pantalla utilizando todos los caracteres gráficos del Spectrum, incluyendo los definidos por el usuario, y guardar los dibujos resultantes, junto con sus características de color.
5) Diseño. Un programa que le permite definir un dibujo de hasta 65536 × 65536 pixels de ancho, añadir y borrar, examinar el dibujo a distintas escalas y girarlo en su totalidad o parte del mismo sobre la pantalla.

3.1 Caracteres

Algunos quizá ya hayan escrito un programa para definir caracteres. Sin embargo, éste es un buen programa de utilidad para aquellos que no lo hayan hecho y constituye una sencilla introducción a algunas de las técnicas que utilizaremos en programas posteriores.

El programa le permite diseñar, a una escala mayor, grupos de hasta cuatro caracteres gráficos, borrar caracteres gráficos ya existentes y guardar en cinta el carácter así creado.

Si no está familiarizado con la idea de los gráficos definidos por el usuario, debería releer el capítulo 14 del manual del Spectrum antes de intentar entrar este programa.

MODULO 3.1.1

```
1000>REM ***********************
1010 REM MENU
1020 REM ***********************
1030 PAPER 7: INK 0: CLS
1040 GO SUB 2400
1050 PRINT "          CARACTERES
"
1060 PRINT '"FUNCIONES DISPONIBL
ES:"
1070 PRINT '"1)INICIALIZAR"
1080 PRINT '"2)CREAR NUEVOS CARA
CTERES"
1090 PRINT '"3)GUARDAR CARACTERE
S"
1100 PRINT '"4)BORRAR CARACTERES
"
1110 PRINT '"5)FINALIZAR"
1120 PRINT '"CUAL REQUIERE?"
1130 INPUT Z$
1140 CLS
1150 IF Z$="1" THEN GO SUB 1230
1160 IF Z$="2" THEN GO SUB 1290
1170 IF Z$="3" THEN GO SUB 2270
1180 IF Z$="4" THEN GO SUB 2040
1190 IF Z$="5" THEN STOP
1200 PAPER 7: CLS
1210 GO TO 1000
1220 STOP
```

Este es un módulo estándar de menú.

MODULO 3.1.2

```
2400>REM ***********************
2410 REM CARGAR CARACTER INICIAL
2420 REM ***********************
2430 RESTORE
2440 FOR I=0 TO 7
2450 READ BYTE
2460 POKE USR CHR$ 144+I,BYTE
```

```
2470 NEXT I
2480 RETURN
2490 DATA 255,129,129,129,129,12
9,129,255
```

Este módulo carga el carácter "□" en la memoria, en la posición ocupada por "A" en el área de gráficos definidos por el usuario. El carácter será utilizado en el curso del programa.

Comentario

Líneas 244∅-247∅. Si se ha releído el capítulo 14 del manual del Spectrum, no debería tener ningún problema para comprender lo que se está haciendo aquí. Los números de las sentencias de DATA representan valores binarios, que, cuando estén guardados en el área de caracteres definidos por el usuario constituirán el carácter "□". El número 255 se representa en binario como 11111111 y el 129 como 1∅∅∅∅∅∅1. Se leen uno a uno y se colocan en la memoria. El CHR$ 144 es "A" en el área definida por el usuario.

Comprobación del módulo 3.1.2

Tras entrar y ejecutar este módulo, el código "A" en modo gráfico deberá visualizar el carácter "□".

MODULO 3.1.3

```
1230>REM ************************
1240 REM VARIABLES
1250 REM ************************
1260 DIM Z(26,8)
1270 LET CARACTERES=1
1280 RETURN
```

Estas son todas las variables.

Comentario

Línea 126∅. Los números que se utilizarán para crear los caracteres definibles por el usuario se guardan en esta tabla para facilitar su manipulación. Cualquier borrado se realiza primeramente en la tabla y después la tabla se carga en la memoria.

MODULO 3.1.4

```
1290 REM ************************
1300 REM DIBUJAR TRAMA
1310 REM ************************
1330 FOR I=2 TO 17
1340 FOR J=6 TO 21
1350 PRINT AT I,J;"□"
1360 NEXT J
1370 NEXT I
1380 FOR I=6 TO 21
1390 PRINT OVER 1; PAPER 5;AT 2,
I;" ";AT 9,I;" ";AT 10,I;" ";AT
17,I;" "
1400 NEXT I
1410 FOR I=2 TO 17
1420 PRINT OVER 1; PAPER 5;AT I,
6;" ";AT I,13;" ";AT I,14;" ";AT
 I,21;" "
1430 NEXT I
1440 PRINT AT 19,0;"5,6,7,8PARA
MOVER□1 PARA MARCAR□0 PARA BORRA
R□9PARA REGISTRAR C."
```

Este módulo dibuja una retícula de 16×16 sobre la cual se construirán los caracteres definibles por el usuario.

Comentario

Líneas 138Ø-143Ø. Hay cuatro caracteres cuadrados separados. Para facilitar la distinción entre ellos, sus bordes están sombreados en azul, utilizando la función OVER para evitar borrar la retícula.

Comprobación del módulo 3.1.4

El módulo debe visualizar una retícula de 16×16 con los bordes de cuatro cuadrados de 8×8 sombreados.

MODULO 3.1.5

```
1450 REM ************************
1460 REM RELLENAR CUADRADOS
1470 REM ************************
1480 DIM A$(4,8,8)
1490 LET X=2: LET Y=6
1500 PRINT AT X,Y; PAPER 8; OVER
 1;"*": PAUSE 5
1510 PRINT AT X,Y; PAPER 8; OVER
 1;"*"
1520 LET T$=INKEY$
1530 IF T$="" THEN GO TO 1500
1540 BEEP .05,40
1550 LET X1=X-1-8*(X>9): LET Y1=
Y-5-8*(Y>13)
```

```
1560 LET XY=1+(Y>13)+2*(X>9)
1570 IF T$="0" THEN LET A$(XY,X1
,Y1)="□"
1580 IF T$="1" THEN LET A$(XY,X1
,Y1)="■"
1590 IF T$="9" THEN GO TO 1660
1600 PRINT PAPER 8;AT X,Y;"□";:
IF A$(XY,X1,Y1)="■" THEN PRINT P
APER 8;AT X,Y;"■"
1610 LET X=X+(T$="6")-(T$="7")
1620 LET X=X+(X<2)-(X>17)
1630 LET Y=Y-(T$="5")+(T$="8")
1640 LET Y=Y+(Y<6)-(Y>21)
1650 GO TO 1500
```

Este módulo es el núcleo del programa. Su propósito es permitir al usuario mover un cursor parpadeante sobre la retícula, coloreando cuadrados o borrando cuadrados que ya estaban coloreados. Cuando el carácter ya está definido a su entera satisfacción, puede salir del módulo y examinar el carácter en su tamaño correcto.

Comentario

Línea 148Ø. Es mucho más fácil manipular el contenido de una tabla que lo que hay sobre la pantalla. Por lo tanto, cuando deba colorearse un cuadrado de la retícula se hará en primer lugar en la tabla A$ y después A$ se visualiza sobre la pantalla. Obsérvese que la tabla A$ tiene en cuenta en sus dimensiones el hecho de que hay cuatro caracteres cuadrados separados.

Línea 149Ø. Estas son las coordenadas de la esquina superior izquierda de la retícula.

Líneas 15ØØ-153Ø. Esta rutina visualiza un cursor parpadeante en la posición X, Y hasta que se pulse una tecla. PAPER se coloca a 8 para que el color del cuadrado quede inalterado. Ya que el asterisco se visualiza dos veces con OVER, no produce ningún efecto sobre el contenido del cuadrado de la retícula.

Línea 154Ø. La función INKEY$ no activa el «bip» por lo que debe sustituirse para indicar que se ha registrado una tecla.

Líneas 155Ø-156Ø. Las coordenadas X, Y se convierten en coordenadas de A$.

Líneas 161Ø-164Ø. Esta es una rutina muy útil a la hora de mover un cursor sobre la pantalla, bajo el control del usuario. La utilización de las condiciones lógicas entre paréntesis modifica el valor de la coordenada X o de la coordenada Y si se pulsa una de las teclas con flecha. Las líneas 162Ø y 164Ø comprueban que el cursor no haya salido fuera de los límites deseados —que en otras circunstancias podrían ser los bordes de la pantalla—. Si los límites se han sobrepasado en algún sentido, el cursor se vuelve atrás utilizando de nuevo una

condición lógica como variable. Estas dos comprobaciones se encontrarán una y otra vez en programas que muevan algo sobre la pantalla.

Comprobación del módulo 3.1.5

Ejecute el programa, inicialícelo, después llame a la función 2 y mueva el cursor parpadeante sobre la retícula, coloreando y borrando según desee.

MODULO 3.1.6

```
1660>REM **********************
1670 REM GRABAR CARACTERES
1680 REM **********************
1690 PAPER 7
1700 INPUT "Es correcta la figur
a?          (ZZZ) para salir ";Q
$
1710 IF Q$="ZZZ" THEN RETURN
1720 IF Q$="S" THEN GO TO 1740
1730 PAPER 5: GO TO 1490
1740 INPUT "CELDAS A GUARDAR? (U
N DIGITO POR CELDA, EN CUALQUIER
 ORDEN):";Q$
1750 LET CELDAS=LEN Q$
1760 IF CARACTERES+CELDAS<=21 TH
EN GO TO 1790
1770 PRINT AT 20,0;"NO HAY SUFIC
IENTE ESPACIO."
1780 RETURN
1790 FOR I=1 TO CELDAS
1800 FOR J=1 TO 8
1810 LET BYTE=0
1820 FOR H=1 TO 8
1830 IF A$(VAL Q$(I),J,H)="■" TH
EN LET BYTE=BYTE+2↑(8-H)
1840 NEXT H
1850 LET Z(CARACTERES+I-1,J)=BYT
E
1860 NEXT J
1870 NEXT I
1890 LET CARACTERES=CARACTERES+C
ELDAS
```

Este módulo transfiere el dibujo creado sobre la retícula a la tabla Z, en forma de una serie de números que, cuando se coloquen en el área de memoria de caracteres definidos por el usuario, reproducirán el carácter exactamente como había sido diseñado.

Comentario

Líneas 179ø-187ø. Cada línea de A$ se trata, en efecto, como un número binario de ocho dígitos, en el que cada cuadrado coloreado

está representado por un "1". El número resultante se coloca en el espacio correspondiente de la tabla Z.

Comprobación del módulo 3.1.6

Puede entrar un carácter y comprobar que en la tabla Z se han entrado los valores adecuados.

MODULO 3.1.7

```
1900>REM ************************
1910 REM ACTIVAR CELDAS
1920 REM ************************
1930 FOR I=1 TO 20
1940 FOR J=0 TO 7
1950 POKE USR CHR$ (144+I)+J,Z(I
,J+1)
1960 NEXT J
1970 NEXT I
1980 INPUT "Desea comprobar los
caracteres? ";Q$
1990 IF Q$<>"S" THEN RETURN
2000 PRINT "PULSE CUALQUIER TECL
A Y LUEGO  PASE A MODO GRAFICO.
"
2010 PAUSE 0
2020 CLS : INPUT Q$
2030 RETURN
```

Los números guardados en la tabla Z se colocan en la memoria de caracteres definidos por el usuario.

Comentario

Líneas 193∅-197∅. Esta es una función similar a la llevada a cabo por el módulo 2.

Línea 2∅2∅. La entrada de un texto en esta línea es meramente una oportunidad para entrar algunos caracteres y comprobar los nuevos caracteres definidos por el usuario que han sido guardados. Este texto no sirve para nada más.

Comprobación del módulo 3.1.7

Ahora ya está en situación de entrar hasta 2∅ de sus propios caracteres y comprobar que se han guardado adecuadamente en la memoria.

MODULO 3.1.8

```
2040 REM ************************
2050 REM BORRAR CARACTERES
2060 REM ************************
2070 IF CARACTERES=1 THEN RETURN

2080 FOR I=2 TO CARACTERES
2090 PRINT CHR$ (143+I)
2100 PRINT ''"DESEA BORRAR ESTE
CARACTER?       (S/N) "
2110 INPUT Q$
2120 IF Q$="S" THEN GO TO 2160
2130 CLS
2140 NEXT I
2150 RETURN
2160 FOR J=I-1 TO CARACTERES-2
2170 FOR K=1 TO 8
2180 LET Z(J,K)=Z(J+1,K)
2190 NEXT K
2200 NEXT J
2210 FOR I=1 TO 8
2220 LET Z(J,I)=0
2230 NEXT I
2240 LET CARACTERES=CARACTERES-1
2250 GO SUB 1900
2260 RETURN
```

Este módulo da al usuario la opción de borrar caracteres que ya habían sido guardados. La operación de borrado realmente se consigue eliminando los elementos correspondientes de la tabla Z y después volviendo a entrar los caracteres en la memoria.

Comprobación del módulo 3.1.8

Intente borrar algunos caracteres y después examinar el resto de los mismos para asegurarse de que no han sido deformados en ninguna forma por el proceso.

MODULO 3.1.9

```
2270 REM ************************
2280 REM GUARDAR CARACTERES
2290 REM ************************
2295 LET Y$="": FOR I=1 TO CARAC
TERES-1: FOR J=1 TO 8: LET Y$=Y$
+CHR$ Z(I,J): NEXT J: NEXT I
2300 DIM X$(LEN Y$): LET X$=Y$
2305 FOR I=1 TO CARACTERES-1
2310 PRINT CHR$ (144+I);" ";
2320 NEXT I
2330 PRINT '"QUE NOMBRE QUIERE D
AR A ESTE     CONJUNTO DE CARACTE
RES?"
2340 INPUT N$: PRINT ''N$
2350 SAVE N$CODE USR "A",21*8
2355 SAVE N$ DATA X$()
```

```
2360 CLS : PRINT ''"A CONTINUACI
ON REBOBINE POR     FAVOR, PONGA
 EN MARCHA LA CINTA Y PULSE CUAL
QUIER TECLA."
2370 PAUSE 0
2380 VERIFY N$CODE USR "A",21*8
2385 VERIFY N$ DATA X$()
2390 PRINT ''"CONJUNTO DE CARACT
ERES        VERIFICADO.": PAUS
E 200: RETURN
```

Al conjunto de caracteres creados por el usuario se le da un nombre y se guarda en el cassette mediante dos formas, una de las cuales está relacionada con el próximo programa de este capítulo.

Comentario

Línea 2295. Los valores de Z se transfieren, en forma de códigos de carácter individuales, a la variable alfanumérica Y$. Ya que el Spectrum no puede guardar en el cassette una variable alfanumérica sin dimensionar como ésta, X$ se dimensiona para que tenga el mismo tamaño que Y$ y se transfiere a X$ el contenido de Y$.

Líneas 23ØØ-234Ø. Esta sección visualiza la totalidad del conjunto de caracteres e invita al usuario a que le dé un nombre.

Línea 235Ø. Esta línea le dice al Spectrum que guarde en el cassette el contenido de su memoria, empezando por el principio del área de caracteres definidos por el usuario e incluyendo 168 octetos de memoria, es decir, la totalidad de la memoria dedicada a los caracteres definidos por el usuario.

Línea 2355. El conjunto de caracteres también se envía al cassette bajo la forma de X$.

Línea 238Ø. El conjunto de caracteres guardados en la cinta puede verificarse de la misma forma que se hace con un programa y sería inteligente el hacerlo, no fuese el caso de que se hubiese producido algún fallo en la grabación de los caracteres tan duramente conseguidos.

Comprobación del módulo 3.1.9

Cree un conjunto de caracteres y guárdelos utilizando este módulo. Asegúrese de que ha guardado el programa, y después desconecte momentáneamente, para borrar el contenido de la memoria. Ahora entre LOAD N$ —utilizando el nombre que le había dado en N$— CODE USR "A", 21*8 y después reproduzca lo que había guardado en el cassette. Cuando la carga haya terminado, deberá des-

cubrir que se han recuperado los nuevos caracteres. Si es así, el programa se ha entrado correctamente y está listo para su utilización.

Resumen

Además de la utilidad que representa la creación de caracteres para utilizarlos en otros programas, en éste se ha encontrado con varias técnicas que volverá a encontrarse de vez en cuando en los programas gráficos. Entre éstas se incluyen el cursor parpadeante, la utilización de las teclas del cursor para mover un carácter sobre la pantalla, la utilización de condiciones lógicas para establecer los límites de este movimiento y la utilización de tablas para simular la pantalla.

Posibles mejoras

1) Intente aumentar el número de caracteres que pueden definirse en un bloque y pasar de 4 a 6.
2) Haga que el programa visualice los caracteres que se están construyendo, en su tamaño correcto, junto a la retícula, a medida que se van construyendo.

3.2 Diccionario

Este pequeño programa aumenta en gran manera la utilidad del generador de caracteres, permitiendo que el usuario pueda crear un diccionario de todos los caracteres definidos por el usuario creados con anterioridad. Entonces el usuario podrá dibujar, basándose en este conjunto de caracteres y crear nuevas combinaciones de caracteres, agrupándolos en conjuntos.

MODULO 3.2.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 INK 0: PAPER 7: CLS : PRINT
 "      DICCIONARIO DE CARACTERES"
1040 PRINT '"1) INICIALIZAR"
1050 PRINT '"2) CARGAR NUEVOS CA
RACTERES"
1060 PRINT '"3) CREAR NUEVO CONJ
UNTO"
1070 PRINT '"4) GUARDAR CONJUNTO
```

```
DE CARACT."
1080 PRINT '"5) FINALIZAR"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1180
1110 IF Z$="2" THEN GO SUB 1430
1120 IF Z$="3" THEN GO SUB 1220
1130 IF Z$="4" THEN GO SUB 1360
1140 IF Z$="5" THEN GO TO 1160
1150 GO TO 1030
1160 INPUT "DESEA REGRABAR? (S/N
)";Q$: IF Q$="S" THEN SAVE "DICC
IONAR": BEEP 1,40: PRINT AT 10,0
;"REBOBINE, Y A CONTINUACION PUL
SE CUALQUIER TECLA PARA VERIFICA
R": PAUSE 0: VERIFY "DICCIONAR":
 PRINT '"PROGRAMA VERIFICADO"
1170 STOP
```

Un módulo estándar de menú.

MODULO 3.2.2

```
1180>REM ***********************
1190 REM INICIALIZAR
1200 REM ***********************
1210 LET P$="": RETURN
```

Es el módulo de inicialización.

Comentario

Línea 121Ø. Esta variable alfanumérica se utiliza para guardar los valores que serán colocados en el área de memoria definida por el usuario.

MODULO 3.2.3

```
1430 REM ***********************
1440 REM CARGAR NUEVOS CARACT.
1450 REM ***********************
1460 INPUT "NOMBRE DEL CONJUNTO
DE          CARACTERES? ";N$
1470 LOAD N$CODE USR "A",168: LO
AD N$ DATA T$()
1480 FOR I=0 TO 20: PRINT CHR$ (
144+I);" ";: NEXT I
1490 INPUT "DESEA ESTE CONJUNTO?
 (S/N)";Q$: IF Q$="N" THEN RETUR
N
1500 LET P$=P$+T$: RETURN
```

Este módulo carga desde la cinta los conjuntos de caracteres creados por el programa anterior y los añade al diccionario.

Comprobación del módulo 3.2.3

Con este módulo entrado, ya podrá cargar conjuntos de caracteres desde la cinta. P$ deberá ser ocho veces el número de caracteres cargado.

MODULO 3.2.4

```
1220>REM ************************
1230 REM IMPRIMIR CARACTERES
1240 REM ************************
1250 FOR I=0 TO 167: POKE USR "A
"+I,0: NEXT I
1260 LET C$=""
1270 FOR I=1 TO LEN P$/160+1: PR
INT AT 0,10;"PAGINA ";I: FOR J=1
 TO 20: FOR K=1 TO 8
1280 IF 160*(I-1)+8*(J-1)+K>LEN
P$ THEN GO TO 1310
1290 POKE USR "A"+(K-1),CODE P$(
160*(I-1)+8*(J-1)+K): NEXT K
1300 PRINT 20*(I-1)+J;" ";CHR$ 1
44: NEXT J
1310 INPUT "NO.CAR.■N=PG. SIG.■Z
ZZ=SALIR";N$: IF N$="N" THEN GO
TO 1350
1315 IF N$="ZZZ" THEN RETURN
1320 LET C$=C$+P$(8*(VAL N$-1)+1
 TO 8*VAL N$): IF LEN C$=168 THE
N PRINT AT 21,0;"CONJUNTO DE CAR
ACTERES LLENO": PAUSE 100: RETUR
N
1330 GO TO 1310
1350 CLS : NEXT I: RETURN
```

Este módulo visualiza el diccionario e invita al usuario a especificar qué caracteres deben incluirse en el nuevo conjunto de caracteres que se está creando.

Comentario

Línea 125Ø. Cualquier carácter definido por el usuario ya existente queda borrado.

Líneas 127Ø-135Ø. En estas líneas hay tres bucles. El bucle I visualiza páginas de 2Ø caracteres; el bucle J visualiza los caracteres individuales que constituyen las páginas, y el bucle K coloca los ocho valores que constituyen cada carácter individual. Cada carácter se coloca en la posición normalmente ocupada por A en el área de gráficos definibles por el usuario y después se visualiza.

Comprobación del módulo 3.2.4

Ahora ya podrá construir nuevos conjuntos de caracteres, partiendo del material proporcionado por el programa anterior.

MODULO 3.2.5

```
1360>REM ************************
1370 REM COLOCAR EN MEMORIA
1380 REM ************************
1390 FOR I=1 TO LEN C$: POKE USR
 "A"+I-1,CODE C$(I): NEXT I
1400 FOR I=0 TO 20: PRINT CHR$ (
144+I): NEXT I
1410 INPUT "NOMBRE CARACT.?";N$:
 SAVE N$CODE USR "A",168
1420 INPUT "REBOBINE Y A CONTINU
ACION ENTER  PARA VERIFICAR.";Q$
: VERIFY N$CODE USR "A",168: PRI
NT AT 21,0;"CONJUNTO DE CARACTER
ES          VERIFICADO.": PAUSE
200: RETURN
```

Este módulo visualiza el conjunto de caracteres, invita al usuario a darle un nombre y después lo guarda como un bloque de códigos que podrán ser utilizados por otros programas según se desee.

Comprobación del módulo 3.2.5

Ahora ya podrá guardar sus conjuntos de caracteres, desconectar el Spectrum y volver a cargar el nuevo conjunto de caracteres. No se olvide de guardar primero el programa.

3.3 Tangram

Este es un programa que pide simplemente que se construya según los propios gustos. Permite al usuario jugar al antiguo juego de figuras chino, llamado Tangram, con dos triángulos pequeños, uno de tamaño medio y dos grandes, junto con un cuadrado y un paralelogramo. Sin embargo, además de esto, el programa es una indicación de cómo pueden dibujarse fácilmente las figuras geométricas, tanto regulares como irregulares, en distintas posiciones u orientaciones, sin el recurso de matemáticas complejas, aunque es más fácil si usted sabe algunas matemáticas.

MODULO 3.3.1

```
1000>REM ***********************
1010 REM VARIABLES
1020 REM ***********************
1030 DEF FN A()=LADO*COS (A*PI/4
)
1040 DEF FN B()=LADO*SIN (A*PI/4
)
1050 LET X=100: LET Y=100
1060 INPUT "BORRAR PANTALLA? (S/
N)";Q$: PAPER 6: INK 1: IF Q$="
S" THEN CLS
1070 LET L$="MOVER CON TEC. DE C
URSOR O ""9"""
1080 LET K$="""0"" PARA ABANDONA
R FIGURA"
1090 LET J$="""1"" GRABAR PANTA
LLA O SALIR."
1100 DIM O$(32)
```

Obsérvese que en este programa no hay menú. Todas las instrucciones se dan en la parte inferior de la pantalla durante la ejecución del programa. Este módulo inicializa las variables necesarias.

Comentario

Líneas 1Ø3Ø-1Ø4Ø. Estas dos sencillas funciones se utilizan para determinar los puntos finales de las líneas que deben dibujarse. Para poder dibujar una línea, utilizando estas dos funciones, tan sólo hace falta saber el punto inicial, el ángulo y la longitud de la línea. Una vez dados estos parámetros, las funciones decidirán las dos coordenadas que serán suficientes para permitir utilizar la orden DRAW. Una mirada a las páginas 68 y 69 del manual de Spectrum demostrarán cómo pueden utilizarse las funciones COS y SIN para conseguirlo. Observe que las funciones pueden manejar únicamente aquellos ángulos que puedan expresarse en unidades de PI÷4 radianes o 1/8 de un círculo. Si desea una mayor flexibilidad podría insertarse un divisor mayor, tal como 18Ø, con lo que se tendrían unidades de un grado.

Línea 1Ø5Ø. Estos son los puntos iniciales a partir de los cuales se realizará cualquier dibujo.

MODULO 3.3.2

```
1650>REM ***********************
1660 REM PIXEL MOVIL
1670 REM ***********************
1680 PRINT AT 20,0;L$'J$
1690 OVER 1: FOR I=1 TO 2: PLOT
BRIGHT 1 AND I=1;X,Y: PLOT X,Y+1
: PLOT X,Y+2: PAUSE 1+2*(I=1): N
EXT I: OVER 0: PLOT INVERSE 1; 0
```

```
VER 1;X,Y
1700 LET T$=INKEY$: IF T$="" THE
N GO TO 1690
1710 IF T$="9" THEN RETURN
1720 IF T$="1" THEN GO TO 1330
1730 IF T$>"9" OR T$<"5" THEN GO
 TO 1690
1740 LET X=X+(T$="8")-(T$="5"):
LET X=X-(X>245)+(X<5)
1750 LET Y=Y+(T$="7")-(T$="6"):
LET Y=Y+(Y<20)-(Y>170)
1760 GO TO 1690
```

Este módulo visualiza un pequeño cursor parpadeante en la posición en la que empezará cualquier figura y permite al usuario mover este cursor sin alterar el contenido de la pantalla. El método es similar al utilizado en el módulo de cursor del primer programa de este capítulo, pero en este caso tan sólo se utilizan tres pixels.

Comentario

Línea 169∅. Obsérvese la utilización de condiciones lógicas para establecer la característica BRIGHT. PLOT INVERSE 1; OVER 1 es una forma de mover la posición de trazado sin trazar un punto. Se utiliza porque al final del bucle, la posición de trazado se ha movido hasta XY + 2. La posición del carácter se pone con BRIGHT para que sea más fácil ver el pequeño cursor.

Líneas 174∅-175∅. Estas líneas son exactamente equivalentes a aquellas que se han utilizado en el último programa para mover el cursor y establecer los límites de su movimiento.

Comprobación del módulo 3.3.2

Debe poder ejecutar el programa y mover el pequeño cursor sobre la pantalla.

MODULO 3.3.3

```
1110>REM ************************
1120 REM SELECCIONAR FIGURAS
1130 REM ************************
1140 GO SUB 1650: PRINT AT 20,0;
O$;K$: LET SUPP=3
1150 INPUT "1:TR 2:PA 3:CU 4:CI
";FIGURA: IF FIGURA=0 THEN GO TO
 1110
1160 IF FIGURA=4 THEN GO SUB 141
0: GO TO 1110
1170 IF FIGURA=2 THEN INPUT "TIP
```

```
0 1 0 2 ";SUPP: IF SUPP=0 THEN G
O TO 1110
1180 IF FIGURA=1 THEN INPUT "TAM
. (1,2,3) ";TAMA: IF TAMA=0 THEN
 GO TO 1110
1190 INPUT "ORIENTACION ";O: LET
 O=3-O: LET A=0: IF O=3 THEN GO
TO 1110
1200 INPUT "VERTICE? ";C1: IF C1
=0 THEN GO TO 1110
1210 LET LARGO=36: IF FIGURA=1 T
HEN LET LARGO=36*((SQR 2)↑TAMA)-
.5
1220 LET CORTO=LARGO/SQR 2
1230 INPUT "DIBUJAR:1 BORRAR:2 "
;DIBUJ: OVER 1
1240 IF FIGURA=1 THEN GO SUB 147
0
1250 IF FIGURA=2 THEN GO SUB 152
0
1260 IF FIGURA=3 THEN GO SUB 157
0
1270 IF DIBUJ=1 THEN INPUT "CONF
IRMAR FIGURA? (S/N) ";Q$: LET A=
0: OVER (Q$="N"): PLOT INVERSE 1
; OVER 1;X,Y
1280 IF DIBUJ<>1 THEN OVER 0: GO
 TO 1110
1290 IF FIGURA=1 THEN GO SUB 147
0
1300 IF FIGURA=2 THEN GO SUB 152
0
1310 IF FIGURA=3 THEN GO SUB 157
0
1320 OVER 0: GO TO 1110
```

Este módulo acepta la información necesaria para visualizar una figura.

Comentario

Línea 115Ø. El programa, tal como está, es capaz de dibujar cuatro figuras básicas: triángulos, paralelogramos, cuadrados y círculos. Si en este módulo se entra un "Ø", dará como resultado la vuelta al cursor parpadeante.

Línea 117Ø. Los paralelogramos no son simétricos, por lo que hay que decirle al programa en qué sentido tiene que dibujarlo. SUPP registra cuál se ha especificado.

Línea 118Ø. Existen tres tamaños de triángulo; son simétricos, rectángulos y cada uno tiene el doble de área que el anterior.

Línea 119Ø. Para cada figura se elige un vértice clave, tal como se muestra en la ilustración. Siguiendo el sentido de las agujas del reloj a partir de este vértice se encuentra el lado clave, y es el ángulo de este lado el que se entra aquí.

Línea 12ØØ. Una vez determinado el ángulo del lado clave, tan sólo falta decir qué vértice de la figura debe situarse sobre el punto

marcado por el cursor. Los vértices se enumeran a partir del vértice clave. Todo esto quizá parezca muy complicado, pero cuando se utilice será evidente.

Línea 121∅. Los lados largos de los triángulos pequeños y de los paralelogramos tienen 36 pixels de largo, una cifra elegida arbitrariamente. Los lados cortos serán por tanto de 36/SQR 2 pixels de largo —discútalo con Pitágoras si no está de acuerdo—. Los lados largos de los triángulos grandes aumentan en múltiplos de dos.

Líneas 123∅-132∅. Esta sección llama a las rutinas que dibujan las figuras, excepto el círculo. Cada rutina se llama dos veces. En la primera ocasión la figura se dibuja sobre lo que ya hay allí (OVER), esto da como resultado el que cualquier línea tocada quede borrada o quede compartida con la nueva figura. Esto todavía permite al usuario ver si la figura está en la posición correcta. Si el usuario lo confirma, la figura se dibuja de nuevo con el OVER puesto a cero, con lo que cualquier error queda validado. Si la figura no se confirma, se dibuja con el OVER de nuevo, con lo que queda borrada. El usuario también puede borrar una figura que tenga un vértice en el punto marcado por el cursor— esto dará como resultado la pérdida permanente de cualquier línea compartida.

Comprobación del módulo 3.3.3

Todavía no puede dibujarse ninguna figura, pero el programa ahora debe aceptar todas las características especificadas con anterioridad.

MODULO 3.3.4

```
1410>REM ***********************
1420 REM CIRCULO
1430 REM ***********************
1440 INPUT "RADIO? ";R: IF R=0 T
HEN RETURN
1450 CIRCLE OVER 1;X,Y,R
1460 INPUT "CONFORME CIRCULO? (S
/N) ";Q$: CIRCLE OVER (Q$="N");X
,Y,R: RETURN
```

Este módulo permite al usuario establecer un radio y dibujar un círculo cuyo origen esté en el punto marcado por el cursor, con la opción de borrarlo de nuevo.

Comprobación del módulo 3.3.4

Debe poder mover el cursor hasta un punto elegido y dibujar un círculo alrededor de este punto.

MODULO 3.3.5

```
1610>REM ************************
1620 REM DIBUJAR FIGURA
1630 REM ************************
1640 DRAW FN A(),FN B(): RETURN
```

Este módulo establece el punto final de una línea que constituirá un extremo de la figura. Las dos funciones se han explicado antes.

MODULO 3.3.6

```
1470>REM ************************
1480 REM TRIANGULOS
1490 REM ************************
1500 LET A$="036": FOR I=0 TO 2:
 LET I1=C1+I-3*((I+C1)>3): LET L
ADO=CORTO: IF I1=2 THEN LET LADO
=LARGO
1510 LET A=O-VAL A$(I1): GO SUB
1610: NEXT I: RETURN
```

Este módulo, cuando se utiliza con el último módulo, dibuja triángulos.

Comentario

Líneas 1500-1510. El método utilizado aquí es sencillo. Para cada figura se crea una cadena de caracteres. Los caracteres individuales de la cadena dicen cuántas unidades de 45° debe girarse cada línea en sentido directo con relación al lado clave. Por lo tanto el lado clave está evidentemente a cero unidades de sí mismo. En el caso de un triángulo, el siguiente lado está a un ángulo de 135°, o 3 unidades, del lado clave. El sencillo bucle de estas dos líneas combina el ángulo entrado para el lado clave con cada uno de los caracteres de la cadena para determinar el ángulo de cada lado. La variable I1 asegura que si el bucle no empieza en 1, lo reducirá a 1 cuando pase de 3. Obsérvese la bifurcación en el bucle causada por la presencia de una sentencia IF.

Comprobación del módulo 3.3.6

Ahora ya podrá especificar y dibujar triángulos.

MODULO 3.3.7

```
1520 REM ************************
1530 REM PARALELOGRAMO
1540 REM ************************
1550 LET A$="0145": FOR I=0 TO 3
: LET I1=C1+I-4*((I+C1)>4): LET
LADO=LARGO: IF I1=2 OR I1=4 THEN
 LET LADO=CORTO
1560 LET A=O-VAL A$(I1)+2*VAL A$
(I1)*(SUPP=2): GO SUB 1610: NEXT
 I: RETURN
```

Este módulo es similar al módulo 6 excepto en que dibuja un paralelogramo. Obsérvese cómo se utiliza SUPP para determinar qué tipo de paralelogramo se dibuja.

MODULO 3.3.8

```
1570 REM ************************
1580 REM CUADRADO
1590 REM ************************
1600 LET LADO=LARGO: FOR I=0 TO
3: LET A=O-2*I: GO SUB 1610: NEX
T I: RETURN
```

Este método visualiza un cuadrado por un método distinto del utilizado en los dos módulos anteriores. En este caso el ángulo del lado clave se va incrementando regularmente y se dibuja un lado a cada incremento. Este método funcionaría para cualquier figura regular y sería algo muy sencillo el permitir que el usuario especificase el número de lados, y el cambio de ángulo para cada lado sería 2Pi/X, donde X es el número de lados.

Comprobación del módulo 3.3.8

Ahora ya debe disponerse de todas las posibilidades para dibujar las figuras del programa.

MODULO 3.3.9

```
1330 REM ************************
1340 REM GUARDAR DIBUJO
1350 REM ************************
```

```
1360 INPUT "QUIERE ABANDONAR? (S
/N)";Q$: IF Q$="S" THEN STOP
1370 INPUT "NOMBRE DEL DIBUJO:";
N$
1380 PRINT AT 20,0;"ARRANCAR PRO
GRAMA CON GO TO 1
                     "
1390 SAVE N$SCREEN$
1400 STOP
```

Este módulo permite guardar en el cassette las figuras creadas, para su uso posterior.

Comentario

Línea 139∅. Esta orden guarda en la cinta el contenido de 6912 posiciones de memoria, que se utilizan para guardar el contenido de la pantalla. Cuando se trata de gráficos en los que se han definido pixels individuales en lugar de caracteres, es el único método práctico de almacenaje.

Comprobación del módulo 3.3.9

Crear un dibujo y guardarlo en cinta. Borrar la pantalla y volver a cargar el dibujo, entrando LOAD "nombre" SCREEN$ y poniendo en funcionamiento la cinta. El dibujo debe irse regenerando gradualmente. Inicializando el programa de nuevo con un GOTO 1, podrá seguir trabajando con el dibujo como si nunca hubiera estado ausente.

Resumen

Si está interesado en realizar dibujos de una forma fácil y bajo su propio control, las técnicas desarrolladas en este capítulo serán de mucha utilidad para usted. Con ellas podrá realizar figuras regulares e irregulares. La técnica de guardar ángulos relativos en una pequeña cadena de caracteres puede utilizarse para definir perímetros bastante complejos, que sería prácticamente imposible analizar matemáticamente. Los módulos para dibujar estas figuras serían, como muestra el programa, cortas y requerirían pocas variables, por lo que son candidatos prácticos para ser añadidos a otros programas que podrían beneficiarse con el dibujo de algunas figuras.

Posibles mejoras

1) En el módulo que dibuja círculos, no se prevé el borrado de un círculo que haya sido dibujado anteriormente. ¿Podría añadir esta característica?
2) Intente añadir un módulo que dibuje un octógono, utilizando el mismo estilo que el módulo que dibuja cuadrados.
3) Modifique el programa de forma que pueda dibujar con cualquier ángulo en lugar de estar limitado a múltiplos de 45°.

3.4 Artista

Este programa, o uno parecido a éste, es una parte importante en el proceso de construir una biblioteca de programas. Si va a necesitar programas que utilicen dibujos como parte de su salida, entonces tendrá que poder crear dibujos de una forma sencilla y poderlos guardar.

La intención de cualquier programa como éste es mejorar las posibilidades del Spectrum, que permite mover el cursor sobre la totalidad de la pantalla y visualizar caracteres gráficos, en cualquier posición de la misma, con la mínima pulsación de teclas. Por último, sería muy interesante en cualquier programa como éste, si el diseño creado pudiera guardarse de una forma más económica que utilizando la función SCREEN$, que requiere cerca de 7ØØØ bytes de memoria y que es bastante lenta a la hora de cargar. Sin embargo, ya que no utilizaremos SCREEN$, tan sólo podremos utilizar caracteres gráficos completos, incluyendo los caracteres definidos por el usuario. El programa se comportará como si los dibujos creados con DRAW, CIRCLE o PLOT no estuvieran allí.

Nota: Si pretende utilizar gráficos definidos por el usuario en cualquier dibujo que deba guardarse, cualquier programa posterior que cargue de nuevo el dibujo deberá realizar la carga con los mismos caracteres.

MODULO 3.4.1

```
1000>REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DIM B$(2,30): FOR I=1 TO 30
: LET B$(1,I)="8": LET B$(2,I)=C
HR$ 0: NEXT I
1040 DIM A$(3,20,30): FOR I=1 TO
 20: LET A$(2,I)=B$(1): LET A$(3
,I)=B$(2): NEXT I
```

```
1050 LET X=1: LET Y=1
1060 LET T=4
1070 DIM O$(32)
1080 PAPER 7: INK 0: CLS
1090 LET MODO=1: DIM L$(2,20): L
ET L$(1)="(MODO 1)": LET L$(2)="
(MODO 2)"
1100 GO SUB 1320
```

Comentario

Línea 1Ø3Ø. Esta línea crea una variable temporal, B$, que se utiliza para acelerar el proceso de la carga de la tabla principal, A$.

Línea 1Ø4Ø. La tabla A$ se utilizará finalmente para guardar tres conjuntos de información:

a) Los caracteres que haya sobre la pantalla.
b) Las características de color de cada uno de los caracteres.
c) Si el carácter tiene que visualizarse con visualización invertida.

La segunda parte de la tabla se carga con ochos, debido a que el código de este carácter, 56, es el mismo que el de un byte de atributo que selecciona Ø para el color de la tinta y 7 para el del papel. Para más explicaciones sobre los códigos de atributos (ATTR) que determinan las características de color, ver la página 116 del manual del Spectrum. La tercera sección de la tabla se rellena con CHR$ Ø para indicar que no hay caracteres invertidos.

Línea 1Ø6Ø. Esta variable registra la dirección en la que se moverá el cursor, expresada en octavos de segmento de un círculo, empezando con un 1 para la dirección vertical hacia arriba.

Línea 1Ø9Ø. MODE es la variable utilizada para guardar si el carácter se visualizará en modo normal o invertido.

MODULO 3.4.2

```
1320 REM ***********************
1325 REM ACEPTA CARACTERES
1330 REM ***********************
1340 PRINT INK 0; PAPER 7;AT 0,0
;L$(MODO);AT 21,0;"ESPACIO W=ATR
B X=FIN Y=GRB Z=MO ";AT 0,21;"DI
RECCION=";T
1350 PRINT INK 0; PAPER 7; OVER
1;AT X,Y;"*": PAUSE 3: PRINT INK
 0; PAPER 7; OVER 1;AT X,Y;"*"
1360 LET T$=INKEY$
1370 IF T$="" THEN GO TO 1350
1380 BEEP .05,12
1390 IF CODE T$>32 AND CODE T$<4
1 THEN LET T=CODE T$-32: GO TO 1
```

```
340
1400 IF CODE T$=64 THEN LET T=2:
 GO TO 1340
1410 IF T$="W" THEN GO SUB 1170:
 GO TO 1320
1420 IF T$="X" THEN STOP
1430 IF T$="Y" THEN GO SUB 1600:
 GO TO 1320
1440 IF T$="Z" THEN LET MODO=MOD
O+1: LET MODO=MODO-2*(MODO>2): G
O TO 1320
1450 IF T$=" " THEN POKE (22528+
32*X+Y),CODE A$(2,X,Y): GO SUB 1
540: GO TO 1340
1460 IF T$="0" THEN PRINT AT X,Y
;" ": LET A$(1,X,Y)=" ": LET A$(
2,X,Y)=CHR$ ATTR (X,Y): GO SUB 1
540: GO TO 1340
1470 IF MODO<>1 THEN GO TO 1510
1480 IF CODE T$>=49 AND CODE T$<
=56 THEN LET T$=CHR$ (CODE T$+80
): LET T$=CHR$ (CODE T$-8*(CODE
T$=136)): GO SUB 1110: LET A$(3,
X,Y)=CHR$ 0: GO SUB 1540: GO TO
1350
1490 IF CODE T$>=65 AND CODE T$<
=85 THEN LET T$=CHR$ (79+CODE T$
): LET A$(3,X,Y)=CHR$ 0: GO SUB
1110: GO SUB 1540: GO TO 1350
1500 IF MODO<>2 THEN GO TO 1350
1510 IF CODE T$>=49 AND CODE T$<
=56 THEN LET T$=CHR$ (191-CODE T
$): LET T$=CHR$ (CODE T$+8*(CODE
 T$=135)): LET A$(3,X,Y)=CHR$ 0:
 GO SUB 1110: GO SUB 1540: GO TO
 1350
1520 IF CODE T$>=65 AND CODE T$<
=85 THEN LET T$=CHR$ (79+CODE T$
): LET A$(3,X,Y)=CHR$ 1: GO SUB
1110: GO SUB 1540: GO TO 1350
1530 GO TO 1320
```

Este módulo distribuye el trabajo entre los módulos que constituyen el resto del programa y convierte los caracteres entrados en caracteres gráficos de acuerdo con el modo seleccionado.

Comentario

Líneas 139Ø-14ØØ. El usuario no tiene que especificar la dirección o mover el cursor cada vez que se entra un carácter. Se mueve automáticamente en una de las ocho direcciones. La dirección se establece pulsando la tecla de Symbol Shift y una de las teclas del 1 al 8.

Línea 145Ø. Si el usuario desea mover el cursor hasta después de un carácter, sin alterar las características de color que están seleccionadas, esto puede hacerse sencillamente entrando un espacio. Las características originales de color de esta posición de carácter, que estaban guardadas en la segunda parte del A$, se colocan de

nuevo en el byte de atributos correspondientes al espacio de este carácter. Espero que ya se haya dado cuenta que el visualizar el cursor dos veces, utilizando la característica OVER, nos asegura que el carácter actual que esté en este espacio quedará inalterado.

Línea 146∅. La entrada de un cero borrará todo lo que hubiera en el cuadrado correspondiente al carácter.

Líneas 148∅-152∅. De acuerdo con el modo seleccionado, el código del carácter correspondiente a la tecla pulsada se convierte en el código del carácter gráfico deseado. Entonces se llama a la rutina que visualiza el carácter y se mueve el cursor. Obsérvese que los caracteres invertidos asociados con las teclas 1-8 no son realmente invertidos. Se trata de otro conjunto de caracteres, tomados del conjunto total de caracteres. Los caracteres invertidos definidos por el usuario sí que son realmente invertidos; es decir, que se visualizan con el color del papel y de la tinta invertidos. Para éstos, se coloca un indicador en la tercera área de A$.

Comprobación del módulo 3.4.2

Con este módulo instalado, deberá ver el cursor parpadeante, las instrucciones abreviadas en la parte inferior de la pantalla y la dirección del cursor en la esquina superior derecha. Podrá cambiar la dirección pulsando la tecla de Symbol Shift y una de las teclas del 1 al 8. Deberá poder cambiar el modo, pulsando la tecla Z. Todavía no podrá entrar ningún carácter.

MODULO 3.4.3

```
1540>REM ***********************
1550 REM INCREMENTO DE POSICION
1560 REM ***********************
1570 LET X=X+(T=4)+(T=5)+(T=6)-(
T=8)-(T=1)-(T=2): LET X=X+(X<1)-
(X>20)
1580 LET Y=Y-(T=6)-(T=7)-(T=8)+(
T=2)+(T=3)+(T=4): LET Y=Y+(Y<1)-
(Y>30)
1590 RETURN
```

Este es un módulo estándar para el movimiento del cursor.

Comprobación del módulo 3.4.3

Ahora ya podrá mover el cursor en la dirección elegida, pulsando la tecla de espacio o el cero.

MODULO 3.4.4

```
1110 REM ************************
1120 REM GRAFICOS ORDINARIOS
1130 REM ************************
1140 LET A$(1,X,Y)=T$
1150 PRINT INVERSE (CODE A$(3,X,
Y)=1);AT X,Y;T$
1160 LET A$(2,X,Y)=CHR$ ATTR (X,
Y): RETURN
```

Este módulo visualiza el carácter deseado en la posición del cursor.

Comentario

Línea 114∅. El carácter que debe visualizarse se coloca en la posición equivalente de la primera parte de A$.

Línea 115∅. El carácter se visualiza invertido si existe el indicador correspondiente en la tercera sección de A$.

Línea 116∅. Las características de color se escogen utilizando la función ATTR y se guardan en la segunda área de A$.

Comprobación del módulo 3.4.4

Ahora ya podrá visualizar cualquier carácter gráfico, en cualquier posición dentro de la retícula de 2∅ × 3∅. No podrá seleccionar características de color hasta que se entre el siguiente módulo.

MODULO 3.4.5

```
1170 REM ************************
1180 REM ATRIBUTOS
1190 REM ************************
1200 PRINT AT 0,0;"ATRIBUTOS    "
1210 DIM Q(4)
1220 FOR I=1 TO 4
1230 INPUT CHR$ (216+I);Q(I)
1240 IF Q(I)>9 THEN GO TO 1230
1250 IF I=1 THEN INK Q(I)
1260 IF I=2 THEN PAPER Q(I)
1270 IF I>2 AND Q(I)>1 THEN GO T
O 1230
1280 IF I=3 THEN FLASH Q(I)
1290 IF I=4 THEN BRIGHT Q(I)
1300 NEXT I
1310 RETURN
```

Este sencillo módulo pide cuatro entradas que establecen las características de INK, PAPER, FLASH y BRIGHT.

Comprobación del módulo 3.4.5

Ahora ya podrá variar las características de color de los distintos puntos de la pantalla.

MODULO 3.4.6

```
1600 REM ***********************
1610 REM CREAR TEXTO EMPAQUETADO
1620 REM ***********************
1630 PRINT INK 0; PAPER 7;AT 21,
0;"REGISTRO
 ": INVERSE 0
1640 LET S$="": LET ESPACIO=0: L
ET IND=1
1650 FOR I=1 TO 20: FOR J=1 TO 3
0
1660 PRINT INK 0;AT I,0;"█"
1670 IF A$(1,I,J)=" " THEN LET E
SPACIO=ESPACIO+1: LET IND=1: GO
TO 1700
1680 IF IND=1 THEN LET S$=S$+CHR
$ 255+CHR$ ESPACIO: LET IND=0: L
ET ESPACIO=0
1690 LET S$=S$+A$(1,I,J)+A$(2,I,
J)+A$(3,I,J)
1700 NEXT J
1710 IF IND=1 THEN LET S$=S$+CHR
$ 255+CHR$ ESPACIO
1720 LET IND=1: LET ESPACIO=0
1730 NEXT I
1740 BEEP 1,2
```

Este módulo crea un texto condensado. Es decir, un texto en el que se han eliminado todos los espacios, a partir del dibujo guardado en A$. El número de espacios eliminados se guarda en forma de un único carácter indicador dentro del texto.

Comentario

Línea 167Ø. Si el bucle encuentra un espacio lo registra en la variable ESPACIO y pone a 1 el indicador IND. Este proceso continúa hasta que no se encuentra ningún otro espacio.

Línea 168Ø. Una vez encontrado un carácter que no es un espacio, se añade un indicador en el texto (CHR$ 255 o COPY) seguido por un único carácter que representa el número de espacios que se han eliminado.

Línea 169Ø. Los tres bytes correspondientes de las tres áreas de A$ se añaden al texto condensado.

Línea 171Ø. El final de cada línea en A$ viene indicado por un COPY seguido por el número de espacios que precedían el final de la línea.

```
1750>REM ************************
1760 REM TEXTO EMPAQUETADO
1770 REM ************************
1780 PAPER 7: CLS : PRINT AT 1,1
;: LET S=0: LET C=1
1790 IF S$(C)=CHR$ 255 THEN LET
C=C+1: IF S$(C)<>CHR$ 0 THEN PRI
NT O$( TO CODE S$(C));: LET S=S+
CODE S$(C): PRINT "  " AND S/30=
INT (S/30);: LET C=C+1: IF C<=LE
N S$ THEN GO TO 1790
1800 IF C>LEN S$ THEN GO TO 1840
1810 IF S$(C)=CHR$ 0 THEN LET C=
C+1
1820 LET S=S+1: INVERSE CODE S$(
C+2): PRINT S$(C);: POKE (23296-
32*(PEEK 23689)-PEEK 23688+32),C
ODE S$(C+1): INVERSE 0: IF S/30=
INT (S/30) THEN PRINT "  ";
1830 LET C=C+3: IF C<=LEN S$ THE
N GO TO 1790
1840 INPUT INK 0; PAPER 7;"GUARD
ARLO? (S/N)";Q$
1850 IF Q$<>"S" THEN RETURN
1860 INPUT "DE POR FAVOR UN NOMB
RE PARA EL  DIBUJO:";N$
1870 DIM P$(LEN S$): LET P$=S$
1880 SAVE N$ DATA P$()
1890 INPUT "REBOBINE LA CINTA, P
ULSE ""ENTER"" Y PONGALA EN MARC
HA PARA VERIFICACION ";Q$
1900 VERIFY N$ DATA P$()
1910 PRINT AT 21,0;"DIBUJO VERIF
ICADO:": STOP
1920 INPUT T$: PRINT CODE T$: GO
 TO 1920
```

Este módulo vuelve a visualizar el texto condensado para su inspección y lo guarda bajo demanda.

Comentario

Línea 179∅. Esta línea examina el texto condensado buscando el indicador especial, COPY, que indica que el siguiente carácter contiene el número de espacios que han sido eliminados. Cuando se encuentra un indicador se inserta el número correcto de espacios. Obsérvese que deben visualizarse dos espacios al final de cada línea para tener en cuenta el hecho de que la retícula original tenía únicamente 3∅ caracteres de ancho.

Línea 182∅. Si el carácter en la posición C no es un indicador, entonces se visualiza con los colores invertidos si el carácter siguiente más 1, C+2, es CHR$ 1. Las características de color se seleccionan colocando (POKE) el valor del carácter C+1 en el área de atributos.

Línea 183∅. El puntero se mueve tres espacios ya que cada carácter requiere tres espacios en el texto condensado.

Líneas 184∅-191∅. Si el usuario desea guardar el dibujo tal como está guardado en el texto condensado, deberá darle primeramente un nombre. Hay que tener en cuenta que el Spectrum no guardará textos adecuadamente si no se han dimensionado, por lo que deberemos crear un nuevo texto, P$, de la misma longitud que el texto condensado.

Una vez guardado el dibujo, podrá recuperarse de la cinta mediante otro programa y volverse a visualizar mediante una rutina equivalente a la de las líneas 178∅-183∅. Véase también módulo 4.2.5.

Resumen

Utilizando esta serie de tres programas tendrá un conjunto de herramientas que le permitirá mejorar muchos de sus programas y llevar a cabo muchas aplicaciones que no serían posible sin la ayuda de herramientas gráficas efectivas.

Al diseñar sus propios gráficos quizá descubra que unas hojas de papel cuadriculado le ayuden a planear sus intenciones antes de sentarse delante del teclado; descubrirá que de esta manera ahorra mucho tiempo. Si puede encontrar dibujos del tamaño adecuado también le puede ayudar el dibujarlos sobre plástico transparente y después pegarlos sobre la pantalla del televisor. Entonces tan sólo hay que seguir el dibujo trazado.

Posibles mejoras

1) Para guardar mediante este programa un dibujo que ocupe toda la pantalla se siguen necesitando hasta 18∅ caracteres. Si no necesita las características invertidas en sus dibujos, esto puede reducirse en un tercera parte. Si sólo requiere un único color en sus dibujos, tan sólo necesitará un solo espacio para guardar cada carácter del dibujo. En una de las aplicaciones del capítulo de temas educacionales se supone que usted habrá modificado los módulos del texto condensado para que guarden y reproduzcan los caracteres y no los atributos.
2) Diseñe un diccionario para sus dibujos, si tiene memoria suficiente, es decir, un programa que guarde un buen número de dibujos y los visualice bajo demanda o secuencialmente.
3) Podrían añadirse dos modos más, uno que aceptase letras normales del teclado y los números, y el otro que aceptase sus equivalentes invertidos. El problema reside en que no

pueden darse órdenes a menos que se empiecen a excluir teclas. Utilizando el ENTER (CHR$13) como orden para cambiar de modo, podrán utilizarse los dos modos aunque no dar órdenes. ¿Podría adaptar el programa para realizar esto?

3.5 Diseño

Tengo un especial cariño por este programa, sencillamente por que las ideas en las que está basado no son mías: fueron tomadas de un excelente libro, *The Principles of Interactive Computer Graphics,* de William M. Newman y Robert F. Sproull. El motivo de este cariño es que el programa me sirve como recordatorio de lo mucho que queda siempre por aprender sobre los principios de programación y de cuantos campos están esperando ser abiertos a cambio del mínimo coste que representan unos pocos libros. Basado en dos sencillos procedimientos sacados de este libro, este programa le permitirá definir un dibujo de hasta 65536 por 65536 pixels de tamaño, examinar este diseño con distintas escalas y girar la totalidad o parte del mismo sobre la pantalla. Una vez se haya acostumbrado a su utilización será capaz de poderlo utilizar en una gran variedad de aplicaciones donde se necesite cambiar y manipular dibujos de una forma rápida y fácil.

MODULO 3.5.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 INK 0: PAPER 6: CLS : PRINT
 PAPER 2; INK 7;AT 1,10;"DISENO"
1040 PRINT '"ORDENES DISPONIBLES
:"
1050 PRINT "     1)INICIALIZAR PA
NTALLA"
1060 PRINT "     2)ANADIR NUEVAS
LINEAS"
1070 PRINT "     3)ESCALAR/GIRAR"
1080 PRINT "     4)BORRAR LINEAS"
1090 PRINT "     5)FINALIZAR"
1100 INPUT Z$: CLS
1110 IF Z$="1" THEN GO SUB 1190
1120 IF Z$="2" THEN GO SUB 1530
1130 IF Z$="3" THEN : LET BUSCAR
=0: GO SUB 1780
1140 IF Z$="4" THEN : LET BUSCAR
=1: GO SUB 1780
1150 IF Z$="5" THEN GO TO 1170
1160 CLS : GO TO 1000
1170 INPUT "DESEA GUARDAR ESTE D
ISENO? ";Q$: IF Q$="S" THEN SAVE
 "DISENO": PRINT "REBOBINE, PULS
E UNA TECLA PARA VERIFICAR": PAU
```

```
SE 0: VERIFY "DISENO": PRINT "VE
RIFICADO"
1180 STOP
```

Este es un módulo estándar de menú.

MODULO 3.5.2

```
1190>REM ***********************
1200 REM INICIALIZAR
1210 REM ***********************
1220 LET IZQUIERDO=0: LET INFERI
OR=0: LET SUPERIOR=167: LET DERE
CHO=255
1230 LET A$=""
1240 DEF FN A()=256*CODE (A$(I1)
)+CODE A$(I1+1)
1250 DEF FN A$()=CHR$ INT (TX1/2
56)+CHR$ (TX1-256*INT (TX1/256))
+CHR$ INT (TY1/256)+CHR$ (TY1-25
6*INT (TY1/256))
1260 DEF FN B$()=CHR$ INT (TX2/2
56)+CHR$ (TX2-256*INT (TX2/256))
+CHR$ INT (TY2/256)+CHR$ (TY2-25
6*INT (TY2/256))
1270 LET A$=" ": RETURN
```

Las funciones y variables que se definen aquí se discutirán en el curso del comentario del programa.

MODULO 3.5.3

```
1280>REM ***********************
1290 REM DIBUJAR LINEAS
1300 REM ***********************
1310 IF (X1<IZQUIERDO AND X2<IZQ
UIERDO) OR (X1>DERECHO AND Y2>DE
RECHO) OR (Y1>SUPERIOR AND Y2>SU
PERIOR) OR (Y1<INFERIOR AND Y2<I
NFERIOR) THEN LET OUT=1: RETURN
1320 IF Y1>SUPERIOR THEN LET BOR
DE=SUPERIOR
1330 IF Y1<INFERIOR THEN LET BOR
DE=INFERIOR
1340 IF Y1<INFERIOR OR Y1>SUPERI
OR THEN LET X1=X1+(X2-X1)*(BORDE
-Y1)/(Y2-Y1): LET Y1=BORDE
1350 IF Y2>SUPERIOR THEN LET BOR
DE=SUPERIOR
1360 IF Y2<INFERIOR THEN LET BOR
DE=INFERIOR
1370 IF Y2<INFERIOR OR Y2>SUPERI
OR THEN LET X2=X2+(X1-X2)*(BORDE
-Y2)/(Y1-Y2): LET Y2=BORDE
1380 IF X1>DERECHO THEN LET BORD
E=DERECHO
1390 IF X1<IZQUIERDO THEN LET BO
RDE=IZQUIERDO
1400 IF X1<IZQUIERDO OR X1>DEREC
```

```
HO THEN LET Y1=Y1+(Y2-Y1)*(BORDE
-X1)/(X2-X1): LET X1=BORDE
1410 IF X2>DERECHO THEN LET BORD
E=DERECHO
1420 IF X2<IZQUIERDO THEN LET BO
RDE=IZQUIERDO
1430 IF X2<IZQUIERDO OR X2>DEREC
HO THEN LET Y2=Y2+(Y1-Y2)*(BORDE
-X2)/(X1-X2): LET X2=BORDE
1440 IF X1-IZQUIERDO>=0 AND X2-I
ZQUIERDO>=0 AND X1-IZQUIERDO<=25
5 AND X2-IZQUIERDO<=255 AND Y1-I
NFERIOR>=0 AND Y1-INFERIOR<=167
AND Y2-INFERIOR>=0 AND Y2-INFERI
OR<=167 THEN PLOT X1-IZQUIERDO,Y
1-INFERIOR+8: DRAW INT (X2-X1),I
NT (Y2-Y1)
1450 RETURN
```

La función de este módulo es tomar dos conjuntos de coordenadas, X1/Y1 y X2/Y2, y decidir si algún trozo de la línea dibujada entre dos puntos así definidos pasará a través de la pantalla. Si cualquier parte de la línea cae dentro de la pantalla, será dibujada, ignorándose las otras partes de la línea.

Comentario

Línea 131∅. La pantalla forma una ventana sobre la totalidad del dibujo que se va creando y los límites del área limitada por la pantalla se guardan en las variables SUPERIOR, INFERIOR, IZQUIERDO y DERECHO. Si INFERIOR se pone a 5∅∅ e IZQUIERDO a 5∅∅ la pantalla visualizará los pixels que caen entre las coordenadas 5∅∅ y 755 horizontalmente y 5∅∅ y 667 verticalmente. El propósito de esta línea del programa es desconsiderar cualquier línea del dibujo cuyo principio y final estén por encima, por debajo o a un lado del área del dibujo cubierto por la pantalla.

Líneas 132∅-133∅. Si una línea empieza por encima o por debajo del área cubierta por la pantalla, estas dos líneas colocan la variable BORDE para que coincida con la parte superior o inferior de la pantalla.

Línea 134∅. Esta línea calcula la posición horizontal por la que la línea pasará sobre el extremo superior o inferior para aquellas líneas que empiecen por encima o por debajo de la pantalla. La fórmula de la primera mitad de la línea no dice nada más que, si por ejemplo, la línea en cuestión pasa a través del borde superior de la pantalla por la mitad de su componente vertical, también pasará por la mitad de su componente horizontal. Evidentemente esto tan sólo será verdad para líneas rectas.

Líneas 135∅-143∅. El mismo procedimiento se lleva a cabo con respecto a las coordenadas Y1, X2 e Y2.

Línea 144∅. Ya que es posible que una línea no esté totalmente por encima, por debajo o a algún lado de la pantalla y no obstante no pase a través de la propia pantalla, esta línea del programa hace una comprobación final de que las coordenadas calculadas están realmente dentro de la pantalla. Si así es, entonces se traza el primer conjunto de coordenadas y se dibuja una línea hasta el segundo.

Comprobación del módulo 3.5.3

Si este módulo funciona correctamente ya podrá ejecutar el programa (RUN) e inicializar las variables. Una vez hecho esto, detenga el programa, y entre en modo directo, 127/2∅∅ y 127/1∅∅ como valores de X1/Y1 y X2/Y2. Con estos cuatro valores, si ejecuta GOTO 128∅, deberán dar como resultado el dibujo de una línea desde la mitad del límite superior de la pantalla hasta más o menos la mitad de la misma.

MODULO 3.5.4

```
1530>REM ************************
1540 REM CURSOR
1550 REM ************************
1560 LET ANGULO=0: LET S=1: GO S
UB 1820
1570 PRINT AT 21,18;"0SAL1DEF2MO
VER";AT 21,0;"X1=";O1;AT 21,8;"Y
1=";O2: LET X=O1: LET Y=O2
1580 GO SUB 1660: IF T$="2" THEN
 GO SUB 1820: PRINT AT 21,18;"0S
AL1DEF2MOVER": LET X=O1: LET Y=O
2: GO TO 1570
1590 LET TTX1=X: LET TTY1=Y: IF
T$="0" THEN RETURN
1600 PRINT AT 21,0;"X2=";X;"  ";
AT 21,8;"Y2=";Y;"  ": PAUSE 50:
GO SUB 1660: IF T$="2" THEN GO S
UB 1820: PRINT AT 21,18;"0SAL1DE
F2MOVER": LET X=O1: LET Y=O2: GO
 TO 1600
1610 IF T$="0" THEN RETURN
1620 LET X2=X: LET TX2=X: LET Y2
=Y: LET TY2=Y: LET X1=TTX1: LET
Y1=TTY1: LET TX1=X1: LET TY1=Y1:
 OVER 1: GO SUB 1280
1630 INPUT "CONFORME? ";Q$: IF Q
$="S" THEN OVER 0: LET A$=A$+FN
A$()+FN B$()
1640 GO SUB 1280: OVER 0: PRINT
AT 21,0;"X1";AT 21,8;"Y1": IF Q$
<>"S" THEN PLOT OVER 1;TX1-IZQUI
ERDO,TY1-INFERIOR+8: PLOT OVER 1
;TX2-IZQUIERDO,TY2-INFERIOR+8
1650 GO TO 1580
1660 PLOT BRIGHT 1; OVER 1;X-IZQ
UIERDO,Y-INFERIOR+8: PAUSE 2: PL
OT BRIGHT 0; OVER 1;X-IZQUIERDO,
Y-INFERIOR+8: LET T$=INKEY$: IF
```

```
T$="" THEN GO TO 1660
1670 LET X=X+(T$="8")-(T$="5")
1680 LET X=X+10*(T$="I")-10*(T$=
"T")
1690 LET Y=Y+(T$="7")-(T$="6")
1700 LET Y=Y+10*(T$="U")-10*(T$=
"Y")
1710 IF X>DERECHO THEN LET X=DER
ECHO
1720 IF X<IZQUIERDO THEN LET X=I
ZQUIERDO
1730 IF Y>SUPERIOR THEN LET Y=SU
PERIOR
1740 IF Y<INFERIOR THEN LET Y=IN
FERIOR
1750 IF T$="1" THEN PLOT OVER 1;
X-IZQUIERDO,Y-INFERIOR+8: RETURN

1755 IF T$="0" OR T$="2" THEN RE
TURN
1760 PRINT AT 21,3;X;"    ";AT 21,
11;Y;"  "
1770 GO TO 1660
```

El propósito de este módulo es permitir al usuario mover un pequeño cursor sobre la pantalla para poder establecer las coordenadas inicial y final de una línea.

Comentario

Línea 156∅. El programa es capaz de ampliar la totalidad del dibujo con cualquier factor especificado y girarlo, aunque las líneas tan sólo pueden entrarse con el dibujo a tamaño normal y sin estar girado. Esta línea establece el ángulo del dibujo (ANGULO) y el factor de reducción (S) antes de llamar al módulo que dibuja la parte del dibujo que el usuario enmarca con la pantalla.

Línea 157∅. Quizás haya observado que la pantalla tiene tan sólo 168 pixels de alto, en lugar de los 176 permitidos. Esto se hace para permitir que la línea 21 sea utilizada para visualizar las coordenadas actuales del cursor con respecto a la esquina inferior izquierda del dibujo total.

Línea 159∅. En la ejecución del programa se utilizan cinco conjuntos de variables en distintos puntos para guardar los mismos datos, y cuyos nombres son X/Y, X1/Y1, X2/Y2, TX1/TY1 y TTX1/TTY2. La sencilla razón de hacerlo es que a veces el valor de una coordenada se modifica temporalmente para algunos propósitos. La T es un indicador de que la variable correspondiente es un lugar de almacenaje temporal. Esta línea también permite al usuario mover la ventana de la pantalla tras definir el principio de una línea. De esta forma pueden definirse líneas que pasen por un área superior a la de una pantalla.

Línea 163∅. Una vez dibujada la línea especificada, se invita al

usuario a que la confirme o que la rechace. Si se confirma, las coordenadas X1/Y1 y X2/Y2 serán guardadas en forma de dos bytes mediante las funciones FN A$ y FN B$, en el texto no definido A$.

Líneas 166Ø-177Ø. Reconocerá esto como una rutina estándar del movimiento del cursor. La única diferencia es que además de las propias teclas del cursor, las teclas que están inmediatamente debajo de ellas y a la derecha (TYUI) pueden utilizarse para mover el cursor 1Ø espacios a la vez, acelerando así el proceso. Si se pulsa la tecla 1 se produce el retorno a una parte anterior del módulo, definiendo así uno de los conjuntos de coordenadas.

Comprobación del módulo 3.5.4

Entrando temporalmente la línea 182Ø con un RETURN y definiendo Ø1 y Ø2 en modo directo como 128 y 83 respectivamente, podrá llamar a este módulo y mover el cursor sobre la pantalla, definir dos posiciones —todavía no podrá mover la pantalla entre el punto inicial y final de la línea— y ver visualizada la línea para que la confirme o la rechace.

MODULO 3.5.5

```
1780>REM ***********************
1790 REM GIRAR
1800 REM ***********************
1810 INPUT "ANGULO?",ANGULO: LET
 ANGULO=ANGULO*PI/180: INPUT "FA
CTOR DE ESCALA?",S: IF S=0 THEN
LET S=1
1820 INPUT "COORDENADAS DEL ORIG
EN?"'O1,O2: IF Z$="2" AND O1<127
 THEN LET O1=127
1830 IF Z$="2" AND O2<83 THEN LE
T O2=83
1840 LET IZQUIERDO=O1-127: LET I
NFERIOR=O2-83: LET DERECHO=IZQUI
ERDO+255: LET SUPERIOR=INFERIOR+
167
1850 CLS
1860 FOR I=2 TO LEN A$ STEP 8
1870 IF I>LEN A$ THEN LET BUSCAR
=0: GO TO 1970
1880 LET I1=I: LET X1=(FN A()-O1
)/S: LET I1=I1+2: LET Y1=(FN A()
-O2)/S: LET I1=I1+2: LET X2=(FN
A()-O1)/S: LET I1=I1+2: LET Y2=(
FN A()-O2)/S
1890 LET TX1=X1: LET TX2=X2: LET
 TY1=Y1: LET TY2=Y2
1900 LET X1=O1+INT (TX1*COS ANGU
LO+TY1*SIN ANGULO)
1910 LET X2=O1+INT (TX2*COS ANGU
LO+TY2*SIN ANGULO)
1920 LET Y1=O2+INT (-TX1*SIN ANG
ULO+TY1*COS ANGULO)
1930 LET Y2=O2+INT (-TX2*SIN ANG
```

```
ULO+TY2*COS ANGULO)
1940 IF BUSCAR=1 THEN OVER 1
1950 GO SUB 1280: IF BUSCAR=1 TH
EN GO SUB 1460: IF T$="0" THEN L
ET T$="": LET BUSCAR=0: RETURN
1960 NEXT I: LET BUSCAR=0
1970 INPUT "ENTER=CONTINUAR/""CC
C=COPY";Q$
1980 IF Q$="CCC" THEN COPY
1990 RETURN
```

Este es el módulo que permite mover sobre el dibujo la ventana representada por la pantalla.

Comentario

Línea 182Ø. La posición de la pantalla con respecto a la totalidad del dibujo se define estableciendo la posición del centro de la pantalla. Obsérvese también que cuando la función del programa es 2 tan sólo están disponibles para el usuario posiciones con coordenadas positivas, es decir, el usuario tan sólo puede dibujar líneas en partes del dibujo que tengan direcciones positivas. Esto es debido a que las direcciones negativas no pueden guardarse en A$ mediante las dos funciones FN A$ y FN B$. En otros momentos durante la ejecución del programa, como por ejemplo cuando un dibujo se ha girado, podrán crearse líneas cuyos extremos tengan coordenadas negativas y entonces se visualizarán sin problemas si la ventana de la pantalla se coloca para que las visualice.

Línea 184Ø. Se establecen los límites de la pantalla para que concuerden con el centro especificado de la misma.

Líneas 186Ø-188Ø. Utilizando la función FN A, se convierten los valores guardados en A$, de nuevo en coordenadas numéricas. Se convierten en posiciones relativas al centro de la pantalla. Entonces esta distancia se multiplica por el factor de escala. Después las coordenadas se giran respecto al centro de la pantalla el ángulo requerido.

Líneas 189Ø-193Ø. El proceso para hacer girar un punto de coordenadas X e Y un ángulo, por ejemplo A, es aplicar la fórmula: X2=X*COS A + Y*SIN A e Y1=−X*SIN A + Y*COS A.

Línea 194Ø. La variable BUSCAR se utiliza para indicar que debe llamarse al módulo de borrado.

Comprobación del módulo 3.5.5

Ahora ya podrá mover la ventana de la pantalla sobre el dibujo y también mover la pantalla entre el primer y segundo conjunto de coor-

denadas al definir una línea. También podrá visualizar la totalidad del dibujo, o parte del mismo, a distintas escalas y a distintos ángulos.

MODULO 3.5.6

```
1460>REM ************************
1470 REM BORRAR LINEAS
1480 REM ************************
1490 PRINT OVER 0;AT 21,0;"YCONF
IRMARNBORRAROSALIR": LET OUT=0:
OVER 1: GO SUB 1280: GO SUB 1280
: LET T$=INKEY$: IF OUT=1 THEN G
O TO 1510
1500 IF T$<>"O" AND T$<>"Y" AND
T$<>"N" THEN GO TO 1490
1510 OVER 0: IF T$="Y" OR OUT=1
THEN GO SUB 1280: RETURN
1520 OVER 1: GO SUB 1280: PAUSE
50: OVER 0: LET A$=A$( TO I-1)+A
$(I+8 TO ): LET I=I-8: RETURN
```

Este módulo dibuja la línea indicada por la variable del bucle I del módulo anterior. La línea se dibuja dos veces con el atributo OVER y entonces se da al usuario la oportunidad de especificar que bien la línea permanezca o que el programa salga del módulo o que la dirección de la línea correspondiente se elimine de A$. La línea parpadeará hasta que se realice una de estas elecciones.

Comprobación del módulo 3.5.6

Ahora ya podrá borrar líneas.

Resumen

Con un poco de imaginación, este programa puede ser una herramienta de mucha utilidad para muchas aplicaciones. Pueden realizarse planos, dibujar mapas o sencillamente jugar. De hecho se puede conseguir que este programa simule muchas de las características de otros ordenadores gráficos mucho más caros, utilizados por ingenieros y científicos en muchos campos. Pero no olvide que el programa es también un ejemplo de una técnica fácilmente accesible, aplicada al Spectrum. Los libros están a nuestro alcance, llenos de potentes ideas que nos ayudarán a liberar la potencia del microordenador.

Posibles mejoras

1) ¿Podría combinar este programa con las técnicas de dibujo de figuras que vimos en el programa Tangram, permitiendo especificar en A$ las direcciones iniciales de algunas figuras frecuentes?
2) El programa sería más flexible si se pudieran visualizar textos como parte del dibujo total. Una vez más, las coordenadas tendrían que poderse guardar en A$.

4. Educación fácil. El Spectrum como tutor doméstico

En este capítulo veremos tres programas que permiten al Spectrum realizar su contribución en el campo de la educación doméstica. El primero de ellos, Respuesta Múltiple, es un programa diseñado para permitir que el usuario entre una serie de preguntas y respuestas, que serán utilizadas como base para la generación de tests aleatorios de respuesta múltiple. El segundo programa, Aprender a Leer, es un programa para enseñar a leer a los niños, y por último Geografía, que ayuda a aprender la situación de las ciudades de cualquier país del mundo que hayan sido entradas en el programa.

El objetivo de estos programas es darle alguna idea de lo que puede conseguirse en este campo sin demasiado esfuerzo. No obstante, a menos que pretenda comprar un conjunto de programas en cassette, con programas especializados dedicados a temas individuales y que incluyen sus propios archivos de datos, la utilidad de estos programas educacionales dependerá siempre de la cantidad de trabajo que esté dispuesto a realizar al entrar los datos en ellos. El mejor programa de generación de preguntas de respuesta múltiple del mundo no es de mucha utilidad si en un determinado momento no está usted dispuesto a sentarse y entrar un número de preguntas suficientes para que sea interesante.

Si está dispuesto a darles a estos programas los datos necesarios con los que trabajar, muchas veces pueden obtener un éxito espectacular por el simple motivo de que funcionan a la velocidad marcada por el alumno, no muestran signos de impaciencia, no dan ningún tipo de penalización por el hecho de hacer trampas o de fallar y están siempre dispuestos para un nuevo intento a cualquier hora del día o de la noche.

4.1 Respuesta múltiple

Este programa es uno de mis favoritos. Cuando lo escribí quedé satisfecho por el hecho de que constituía un trabajo competente que haría la tarea para la que estaba diseñado. Hasta que entré un grupo de preguntas y respuestas y lo probé con otras personas, no me di

cuenta que tales programas hacían del aprendizaje algo tan apasionante como cualquier otro juego.

Al igual que el programa Archivo, éste tiene también propiedades de camaleón, y está diseñado para cambiar su color para adecuarse a sus necesidades. En un momento determinado quizá desee que sea un profesor de francés, ofreciendo una variedad de palabras en francés como posibles traducciones de una palabra en español. Más tarde quizá lo convierta en un programa que plantee complejas preguntas sobre la historia del siglo XIX, dando una serie de fechas como posibles respuestas. El objetivo del programa es permitirle que pueda hacer todo esto y mucho más sin tener que hacer cambios en el propio programa.

El programa también es interesante por la forma en que guarda los datos. El programa Archivo, como recordará, guardaba los datos en un único archivo bastante largo, controlando cada registro mediante una tabla de punteros. El programa de Respuesta Múltiple guarda sus datos en una serie de tablas bidimensionales, es decir, que A$ (1ØØØ,1Ø) proporcionaría 1ØØØ espacios de 1Ø caracteres de largo. Esto tiene la desventaja de que es imposible utilizar al máximo el espacio, como se indicó anteriormente. Sin embargo, en este caso el usuario puede elegir la forma en que serán guardados los elementos, y si un elemento parece que va a ser demasiado largo en comparación con el resto, entonces puede abreviarse en lugar de aumentar el tamaño de la tabla en donde deben colocarse. Si el problema del ahorro de espacio no es crucial, entonces las tablas bidimensionales tienen la ventaja de que cada elemento tiene un lugar claramente identificable dentro del archivo, lo que hace más fácil la búsqueda de los datos.

Sin embargo, todavía queda un problema, y es que una tabla puede ser un inconveniente a la hora de borrar o insertar elementos en mitad de la misma, ya que todos los elementos sucesivos deberán desplazarse un lugar para cubrir el hueco resultante o para dejar espacio para el nuevo registro. Ya que no existe una orden que mueva un bloque completo dentro de una tabla, esto tendrá que hacerse elemento a elemento, lo que es un proceso que necesita cierto tiempo.

Podemos resolver el problema que se refiere a la inserción de elementos no colocando los nuevos elementos en mitad de la tabla, en su lugar correcto, si no insertándolos en el primer espacio disponible y dejando que una tabla de punteros se encargue de decirnos el orden en que deben tomarse los elementos.

El borrado no es tan sencillo. El programa no será de demasiada utilidad si no se pueden eliminar los elementos que sean incorrectos y, se mire como se mire, si se elimina el primer elemento de la tabla quedará una línea vacía. Muy pronto, si se necesita borrar elementos con cierta regularidad, terminará teniendo una tabla llena de espacios

vacíos y sin espacio al final para colocar los nuevos elementos de datos.

La solución adoptada aquí es mantener un registro de la posición de cualquier elemento que haya sido borrado del archivo y utilizar este registro como indicador de dónde deben colocarse los nuevos registros. En otras palabras, si el último elemento borrado estaba en la posición uno, el siguiente elemento que vamos a entrar se colocará en la posición uno. Si, como a menudo sucederá, no quedan agujeros en la tabla, el programa colocará el nuevo elemento al final de los elementos actuales. Es sencillo de comprender cuando se capta la idea, y, como verá cuando entre el programa, no es difícil en la práctica.

MODULO 4.1.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 BORDER 1: INK 0: PAPER 7: C
LS : PRINT "        Respuesta Mul
tiple"
1040 PRINT '"ORDENES DISPONIBLES
:"
1050 PRINT '"1)INICIALIZAR"
1060 PRINT '"2)ENTRAR NUEVOS ELE
MENTOS"
1070 PRINT '"3)BUSCAR/ELIMINAR"
1080 PRINT '"4)ENTRAR NUEVOS TIP
OS"
1090 PRINT '"5)GENERAR PREGUNTAS
"
1100 PRINT '"6)VER O PONER A CER
O PUNTUACION"
1110 PRINT '"7)FINALIZAR"
1120 INPUT "Cual requiere?";Z$
1140 CLS
1150 IF Z$="1" THEN GO SUB 1270
1160 IF Z$="2" THEN GO SUB 1800
1170 IF Z$="3" THEN GO SUB 2460
1180 IF Z$="4" THEN GO SUB 1670
1190 IF Z$="5" THEN GO SUB 2880
1200 IF Z$="6" THEN GO SUB 3380
1210 IF Z$="7" THEN GO TO 1240
1220 CLS
1230 GO TO 1000
1240 PRINT AT 7,7;"Respuesta Mul
tiple"
1250 INPUT "Ha entrado nueva inf
ormation que quiera guardar? (S/
N)";Q$
1260 IF Q$="S" THEN SAVE "REMULT
": BEEP 1,40: PRINT AT 12,0;"REB
OBINE, A CONTINUACION ""ENTER""
PARA VERIFICAR.": PAUSE 0: VERIF
Y "REMULT": PRINT '"VERIFICADO"
1265 STOP
```

Este es un módulo estándar de menú.

```
1270>REM ***********************
1280 REM INICIALIZAR
1290 REM ***********************
1295 DIM O$(32)
1300 DIM C$(2,20)
1310 LET I$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 2
1320 DIM C(2)
1330 LET L$="███████████████████
█████████████"
1340 LET ELEMENTOS=2
1350 LET P$="    "
1360 LET ELEMFIN=3
1370 LET BIEN=0
1380 LET QTOT=0
1390 INK 1: PRINT "            PRO
FESOR"
1400 PRINT '"NOMBRE DE LA RESPUE
STA:";
1410 INPUT C$(1)
1420 PRINT C$(1)
1430 PRINT '"LONG. DE LA RESP. M
AS LARGA:";
1440 INPUT C(1)
1450 LET C(1)=C(1)+1
1460 PRINT C(1)-1
1470 PRINT ''"NOMBRE DE LA PREGU
NTA:";
1480 INPUT C$(2)
1490 PRINT C$(2)
1500 PRINT '"LONG. DE LA PREG. M
AS LARGA:";
1510 INPUT C(2)
1520 PRINT C(2)
1530 INPUT "Son correctas? (S/N)
";Q$
1550 CLS
1560 IF Q$<>"S" THEN GO TO 1270
1570 DIM F$(C(1))
1580 DIM G$(C(2))
1590 LET C1=C(1)+C(2)
1600 LET C2=INT (25000/C1)
1610 DIM A$(C2,C(1))
1620 DIM B$(C2,C(2))
1625 LET A$(1,1)=CHR$ 0
1630 LET A$(2,1)=CHR$ 255
1640 LET TIPOS=0
1650 DIM D$(10,20)
1660 DIM D(2,10)
1670 PRINT "ENTRAR TIPOS ";"EXTR
A " AND (TIPOS>0);"DE RESPUESTA:
"
1680 PRINT '"(""ZZZ"" PARA SALIR
)"
1720 INPUT Q$
1730 IF Q$="ZZZ" THEN RETURN
1740 LET D$(TIPOS+1)=Q$
1750 PRINT TIPOS+1;") ";D$(TIPOS
+1)
1755 LET TIPOS=TIPOS+1
1760 IF TIPOS<10 THEN GO TO 1720
1770 PRINT '"SOLO SE PERMITEN 10
 TIPOS."
1780 PRINT '"Pulsar una tecla pa
ra continuar."
1785 PAUSE 0
1790 RETURN
```

Este módulo inicializa las variables y redimensiona las tablas para adecuarlas a nuevas aplicaciones del programa.

Comentario

Líneas 139∅-158∅. Los datos se guardan en el programa con los títulos de preguntas, respuestas y tipos, siendo el usuario quien define los nombres que se dan a cada uno de ellos. Si el examen va a tener la forma de una palabra en español acompañada de cinco posibles traducciones en francés, se podría entrar «Francés» como respuesta de «nombre de la respuesta» y «España» como respuesta de «nombre de las preguntas». También tendrá que decidir la longitud máxima de las preguntas y de las respuestas para cada aplicación determinada.

Líneas 159∅-162∅. Las dos tablas principales en las que se guardarán los datos (A$=respuestas, B$=preguntas) se redimensionan para que ocupen la totalidad del espacio de memoria disponible, independientemente de la longitud de las líneas de cada una. Cuanto más cortos sean los elementos más líneas habrá.

Líneas 165∅-179∅. A cada elemento se le puede dar un tipo, por ejemplo nombre, verbo, etc. en el caso de un examen de Francés. Estos tipos pueden utilizarse para después hacer el examen más estricto. Se pueden especificar hasta 1∅ tipos y sus nombres se guardarán en D$. La tabla D se utilizará para guardar cuántos elementos de cada tipo hay y dónde están agrupados dentro del archivo. Obsérvese que se puede prescindir de los tipos entrando ZZZ antes de entrar cualquier nombre de tipo. En este caso el programa ya no hará ninguna referencia a los tipos.

Comprobación del módulo 4.1.2

Cuando se ejecute, este módulo deberá pedir títulos y tipos.

MODULO 4.1.3

```
1800>REM ************************
1810 REM ENTRADA DE NUEVOS
         ELEMENTOS
1820 REM ************************
1830 PRINT "            NUEVOS ELEM
ENTOS"
1840 PRINT PAPER 6;'C$(1)
1850 INPUT F$
1860 IF ELEMENTOS<>C2 THEN GO TO
 1890
1870 PRINT '"NO HAY SITIO PARA M
```

```
AS ELEMENTOS."
1880 GO TO 2130
1890 IF F$(1 TO 3)="ZZZ" THEN GO
 TO 2130
1900 PRINT INVERSE 1;F$
1910 PRINT PAPER 6;'C$(2)
1920 INPUT G$
1930 PRINT INVERSE 1;G$
1940 IF D$(1,1)<>" " THEN GO TO
1970
1950 LET T=0
1960 GO TO 2040
1970 PRINT '"TIPOS:"
1980 FOR I=1 TO TIPOS
1990 PRINT I;") ";D$(I)
2000 NEXT I
2010 INPUT T
2030 PRINT '"TIPO SELECIONADO:";
D$(T)
2040 INPUT "Son correctos? (S/N)
";Q$
2060 CLS
2070 IF Q$<>"S" THEN GO TO 1830
2075 IF T<>0 THEN LET D(1,T)=D(1
,T)+1
2080 LET F$=CHR$ T+F$
2090 GO SUB 2190
2100 GO SUB 2330
2110 LET ELEMENTOS=ELEMENTOS+1
2120 GO TO 1800
2130 LET TOTAL=2
2140 FOR I=1 TO 10
2150 LET D(2,I)=TOTAL
2160 LET TOTAL=TOTAL+D(1,I)
2170 NEXT I
2180 RETURN
```

Este módulo acepta nuevas entradas con los títulos y tipos especificados en el módulo anterior.

Comentario

Línea 186∅. C2 es el número de líneas de la tabla principal.

Línea 194∅. Al usuario únicamente se le pide que especifique un tipo, si existe al menos un nombre de tipo principal guardado en D$.

Línea 2∅8∅. El tipo —un ∅ si no se han especificado tipos— se añade al principio de la respuesta en forma de un único carácter indicador.

Líneas 213∅-217∅. Este bucle transfiere el total acumulativo de las sumas de la primera columna de la tabla D, es decir, los números de elementos de cada tipo, a la segunda columna de la misma tabla. Ya que los elementos están guardados por orden según el tipo, con esto se guardará efectivamente la dirección inicial de cada grupo con el mismo tipo, en la segunda columna.

Comprobación del módulo 4.1.3

El programa deberá aceptar la entrada de un elemento, aunque no podrá colocarse en la tabla.

MODULO 4.1.4

```
2190 REM ************************
2200 REM BUSQUEDA BINARIA
2210 REM ************************
2215 DEF FN A()=256*CODE I$(2*S-
1)+CODE I$(2*S)
2220 LET POTEN=INT (LN ELEMENTOS
/LN 2)
2230 LET S=2↑POTEN
2240 FOR I=POTEN-1 TO 0 STEP -1
2250 LET P=FN A()
2260 LET S=S+(2↑I)*(A$(P)<F$)-(2
↑I)*(A$(P)>F$)
2270 IF S<2 THEN LET S=2
2280 IF S>ELEMENTOS-1 THEN LET S
=ELEMENTOS-1
2290 NEXT I
2300 LET P=FN A()
2310 IF A$(P)>F$ THEN LET S=S-1
2320 RETURN
```

Este módulo determina la posición correcta de los nuevos elementos según el orden existente y es equivalente al módulo de búsqueda de Archivo. El método utilizado es también la búsqueda binaria.

MODULO 4.1.5

```
2330 REM ************************
2340 REM INSERCION
2350 REM ************************
2360 IF LEN P$=4 THEN GO TO 2400
2370 LET LUGAR=256*CODE P$(3)+CO
DE P$(4)
2380 LET P$=P$( TO 2)+P$(5 TO )
2390 GO TO 2420
2400 LET LUGAR=ELEMFIN
2410 LET ELEMFIN=ELEMFIN+1
2420 LET A$(LUGAR)=F$
2430 LET B$(LUGAR)=G$
2440 LET I$=I$( TO 2*S)+CHR$ INT
 (LUGAR/256)+CHR$ INT (LUGAR-256
*INT (LUGAR/256))+I$(2*S+1 TO )
2450 RETURN
```

El nuevo elemento se inserta en el primer espacio disponible y su dirección se guarda en la tabla I$.

Comentario

Líneas 236∅-239∅. P$ guarda las direcciones de los huecos en la tabla. Cuando no hay tales huecos la longitud de esta variable alfanumérica es 4; el mínimo para evitar que la línea 238∅ nos dé un mensaje de error. Siempre que P$ sea mayor que cuatro caracteres, el segundo par se convierte en una dirección.

Líneas 24∅∅-241∅. Si no hay huecos, que vendrían indicados por P$, el nuevo elemento se coloca al final del archivo, en el primer espacio vacío.

Líneas 242∅-244∅. La respuesta y la pregunta, F$ y G$, se colocan en la línea LUGAR de las tablas A$ y B$ respectivamente. Obsérvese que LUGAR contiene la dirección actual del elemento en el archivo, y no su posición correcta por orden, que está guardada en la variable S. Esta posición se determina mediante la búsqueda binaria. Ya que el tipo se ha colocado al principio de la respuesta, los elementos están ordenados principalmente con respecto al tipo.

Comprobación del módulo 4.1.5

Entrar una serie de elementos del mismo tipo: deben entrarse en orden alfabético invertido, es decir, e,d,c,b,a. Los elementos deberán colocarse en las tablas en el orden en que se han entrado. Ahorá examine I$. Deberá ser capaz de convertir los pares de caracteres en punteros que señalen las posiciones 5,4,3,2,1 o de cuantos elementos haya entrado en orden alfabético invertido. Recuerde que los elementos se clasifican con respecto a las respuestas.

MODULO 4.1.6

```
2460>REM ************************
2470 REM BUSQUEDA DE USUARIO
2480 REM ************************
2485 IF ELEMENTOS=2 THEN RETURN
2490 PRINT "             BUSQUEDA"
2495 PRINT AT 21,8;"TOTAL DE ELE
MENTOS:";ELEMENTOS-2
2500 PRINT AT 3,0;"ORDENES DISPO
NIBLES:"
2510 PRINT '">""ENTER"" SIGUIENT
E ELEMENTO"
2520 PRINT ">NUMERO POSITIVO O N
EGATIVO PARA MOVER PUNTERO"
2530 PRINT ">""DDD"" ELIMINAR EL
EMENTO"
2540 PRINT ">""ZZZ"" ABANDONAR F
UNCION"
2550 PRINT 'L$
2580 LET S=2
2600 LET P=FN A()
2610 PRINT AT 12,0;"REGISTRO No.
```

```
";S-1
2620 PRINT 'A$(P,2 TO )
2630 PRINT B$(P)
2640 IF CODE (A$(P,1))=0 THEN GO
 TO 2660
2650 PRINT 'D$(CODE A$(P,1))
2660 INPUT S$
2670 IF S$="DDD" THEN GO SUB 278
0: RETURN
2690 IF S$="ZZZ" THEN RETURN
2700 IF S$<>"" THEN GO TO 2740
2710 LET S=S+1
2720 IF S=ELEMENTOS THEN RETURN
2730 GO TO 2600
2740 LET S=S+VAL S$
2750 IF S>ELEMENTOS THEN RETURN
2760 IF S<2 THEN LET S=2
2770 GO TO 2600
```

El propósito de este módulo es permitir al usuario recorrer los registros y borrar los no deseados. Este módulo es más sencillo que el módulo de búsqueda del programa Archivo, pero tiene la posibilidad de visualizar el siguiente registro o saltarse un número especificado de elementos. En realidad no busca.

MODULO 4.1.7

```
2780>REM ***********************
2790 REM ELIMINACIONES
2800 REM ***********************
2810 LET P=FN A()
2815 LET D(1,CODE A$(P,1))=D(1,C
ODE A$(P,1))-1
2820 LET A$(P)=""
2830 LET B$(P)=""
2840 LET I$=I$( TO 2*S-2)+I$(2*S
+1 TO )
2850 LET ELEMENTOS=ELEMENTOS-1
2860 LET P$=P$( TO 2)+CHR$ INT (
P/256)+CHR$ INT (P-256*INT (P/25
6))+P$(3 TO )
2865 GO SUB 2130
2870 RETURN
```

Este módulo elimina elementos del archivo y guarda la dirección del hueco resultante en P$.

Comentario

Líneas 281∅-286∅. Ya que la variable S está señalando al elemento que se está visualizando, se utiliza para extraer una dirección de la tabla de punteros, I$, y las líneas indicadas por esta dirección en A$ y B$ quedan vacías. La dirección se elimina de I$ y se coloca en P$.

Línea 2865. Las direcciones iniciales de los distintos grupos de tipos de reajustan.

Comprobación del módulo 4.1.7

Borre un elemento y compruebe que el resto de los datos siguen estando en el orden correcto.

MODULO 4.1.8

```
2880>REM ************************
2890 REM PREGUNTAS ALEATORIAS
2900 REM ************************
2910 LET PREGUNTA=0
2920 PRINT "               PREGUNTAS"
2930 PRINT '"DESEA QUE LAS RESPU
ESTAS           POSIBLES SE DERIVEN
 DE UN MISMO TIPO? (S/N)"
2940 INPUT Q$: CLS
2960 IF Q$="S" THEN LET PREGUNTA
=1
2970 DIM Q(5)
2980 LET Q1=INT (RND*(ELEMENTOS-
2)+2)
2990 LET S=Q1
3000 LET P=FN A()
3010 LET Q2=INT (RND*5+1)
3020 LET Q(Q2)=P
3030 LET BASE=2
3040 LET NUMERO=ELEMENTOS-2
3045 IF CODE (A$(P,1))=0 THEN GO
 TO 3080
3050 IF PREGUNTA=0 OR D(1,CODE A
$(P,1))<5 THEN GO TO 3080
3060 LET BASE=D(2,CODE A$(P,1))
3070 LET NUMERO=D(1,CODE A$(P,1)
)
3080 FOR I=1 TO 5
3090 IF I=Q2 THEN GO TO 3170
3100 LET S=INT (RND*NUMERO+BASE)
3110 LET P=FN A()
3120 IF P=Q(Q2) THEN GO TO 3100
3130 FOR J=1 TO I
3140 IF P=Q(J) THEN GO TO 3100
3150 NEXT J
3160 LET Q(I)=P
3170 NEXT I
3180 PRINT AT 1,0;C$(2)
3190 PRINT 'B$(Q(Q2))
3200 PRINT 'L$''
3205 PRINT C$(1)''
3210 FOR I=1 TO 5
3220 PRINT I;") ";A$(Q(I),2 TO )
3230 NEXT I
3240 PRINT ''"CUAL RESPUESTA CRE
E CORRECTA?"'"ENTRE EL NUMERO:";
3250 LET QTOT=QTOT+1
3260 INPUT RESPUESTA: PRINT "";R
ESPUESTA
3270 IF RESPUESTA=Q2 THEN GO TO
3300
3280 PRINT '"INCORRECTO. LA RESP
UESTA           CORRECTA ERA:";A$(Q
```

```
(Q2),2 TO )
3290 GO TO 3320
3300 FOR I=14 TO 20
3302 PRINT AT I,0;O$
3304 NEXT I
3306 FLASH 1: PRINT AT 8+RESPUES
TA,3;A$(Q(RESPUESTA),2 TO ): PRI
NT AT 16,10;"CORRECTO": PAUSE 50
: FLASH 0
3310 LET BIEN=BIEN+1
3320 PRINT """ENTER"" nueva preg
unta o ""ZZZ"" para abandonar f
uncion."
3340 INPUT Q$
3350 CLS
3360 IF Q$="ZZZ" THEN RETURN
3370 GO TO 2970
```

Este módulo establece el examen de respuesta múltiple basándose en los elementos guardados.

Comentario

Líneas 288Ø-3ØØØ. Al usuario se le da la opción de poder elegir si las cinco respuestas a cada pregunta deben sacarse de la totalidad del archivo o únicamente de entre elementos del mismo tipo.

Líneas 298Ø-3ØØØ. Se elige aleatoriamente un elemento del archivo.

Líneas 3Ø1Ø-3Ø2Ø. Las direcciones de las cinco respuestas se guardarán en la tabla Q. La dirección de la respuesta correcta se coloca en una posición aleatoria en esta tabla.

Líneas 3Ø3Ø-317Ø. Cuando se crea un número aleatorio dentro de ciertos límites es necesario saber dos cosas: primeramente la base sobre la cual el número se va a construir, y en segundo lugar el rango por encima de esta base dentro del cual debe caer el número aleatorio. La función RND del Spectrum produce un valor seudoaleatorio entre Ø y 1, por lo que un número que tenga la forma de X+INT (Y*RND) será capaz de variar entre X y X+Y−1. El valor mínimo que puede tomar Y*RND puede ser menor que uno, por lo que INT(Y*RND) sería igual a cero. El máximo valor de INT(Y*RND) es Y−1 ya que RND nunca puede ser igual a 1.

En esta sección del programa deberá generarse un cierto número de direcciones aleatorias y esto se conseguirá partiendo de las variables BASE y NUMERO. Si el usuario ha especificado que las respuestas deben elegirse de entre la totalidad del archivo en lugar de entre las de un tipo específico, la BASE se pone a 2, el primer elemento utilizable después del indicador de principio del archivo. NUMERO se coloca igual al número real de elementos, es decir, el número de elementos menos los dos marcadores de principio y final del archivo.

Si el usuario ha especificado únicamente respuestas del mismo tipo, la BASE se pone igual a la primera dirección del grupo sobre el que debe elegirse aleatoriamente y número se pone igual al número de elementos dentro de este grupo. Esto se hace únicamente si existen cinco o más elementos dentro del grupo en cuestión. La respuesta elegida se compara con la respuesta adicional para ver que no sea la misma y después con el resto de las respuestas elegidas aleatoriamente. Si la nueva elección aleatoria no es una duplicación, entonces su dirección se coloca en Q, en una posición aleatoria.

Líneas 318Ø-323Ø. La pregunta se visualiza en la parte superior de la pantalla; la respuesta se visualiza en la parte inferior.

Comprobación del módulo 4.1.8

Entre algunas preguntas y respuestas y examínese usted mismo.

MODULO 4.1.9

```
3380 >REM ************************
3390 REM PUNTUACION
3400 REM ************************
3410 PRINT "          PUNTUACION
"
3420 IF QTOT<>0 THEN GO TO 3470
3430 PRINT ''''"        AUN NO HAY
 PUNTUACION."
3440 PRINT '"Cualquier tecla par
a continuar."
3450 PAUSE 0
3460 RETURN
3470 PRINT '"TOTAL DE PREGUNTAS:
";QTOT
3480 PRINT '"CORRECTAS:";BIEN
3490 PRINT "EVALUACION:";INT (((
BIEN-QTOT/5)/(QTOT*.8))*100);" P
ER CIENTO."
3500 PRINT '"QUIERE PONER A CERO
 LA          PUNTUACION? (S/N)"
3510 INPUT Q$
3520 IF Q$<>"S" THEN RETURN
3530 LET QTOT=0
3540 LET BIEN=0
3550 RETURN
```

Este módulo está diseñado para visualizar la puntuación hasta el momento y borrarla bajo demanda.

Comentario

Línea 349Ø. Esta línea es un intento de dar una evaluación adecuada de la puntuación, teniendo en cuenta el hecho de que la pul-

sación aleatoria de las teclas daría como resultado una puntuación del 20% como promedio. Este 20% hipotético se resta tanto del número total de preguntas como de las respuestas correctas y la evaluación se hace con el resto.

Comprobación del módulo 4.1.9

Si previamente ha comprobado el módulo 8, las variables ya habrán sido establecidas. Al llamar a este módulo se le dará una puntuación y una evaluación.

Resumen

Este es un programa bastante potente, pero recuerde que esto tan sólo podrá confirmarlo si entra suficientes datos para que sea interesante. Además de su utilización como herramienta educacional, también es una demostración sencilla de un método para acelerar el acceso a tablas, eliminando la necesidad de desplazar el contenido del archivo cada vez que se realiza el borrado de un registro.

El núcleo del programa podría aplicarse fácilmente a una gran variedad de utilizaciones en las que la velocidad de acceso a los elementos sea más importante que el ahorrar hasta el último espacio de memoria.

Posibles mejoras

1) El programa tal como está no da ninguna recompensa por las respuestas correctas o por buenas puntuaciones. ¿Podría programar algún tipo de recompensa visual en el punto adecuado?
2) La programación es una tarea más bien solitaria, pero utilizando un programa como éste se podría mejorar mediante la cooperación con otras personas. Convenza a alguien para que se compre este libro y póngalo a construir un archivo sobre física, mientras que usted empieza con el de historia.
3) ¿Podría añadir funciones de búsqueda más potentes que se basasen en la función de búsqueda del programa Archivo?

4.2 Palabras

La moraleja del programa anterior es que cuando se tiene un método que funciona correctamente, hay que adaptarlo para que realice varias tareas, aunque se parezca sospechosamente a otros programas que haya escrito antes. Este programa es una copia del de Respuesta Múltiple cuya única diferencia es que las preguntas están en forma de dibujos. Pretendió ser un programa para enseñar a leer, pero naturalmente es aplicable a cualquier otro tema donde todo tenga que presentarse visualmente y deban realizarse preguntas.

La capacidad del programa es relativamente limitada ya que se ha reservado un espacio de 4ØØ caracteres para cada pequeño dibujo. Los dibujos son los que se han generado con el programa Artista. Esto significa que la cadena de 4ØØ caracteres representa la versión compactada de algo que originalmente tenía un máximo de unos 12Ø caracteres. El dibujo debe utilizar sólo la mitad inferior de la pantalla.

MODULO 4.2.1

```
1000>REM ***********************
1010 REM MENU
1020 REM ***********************
1030 BORDER 1: INK 0: PAPER 6: C
LS
1040 PRINT "            PALABRAS"
1050 PRINT '"ORDENES DISPONIBLES
:"
1060 PRINT '"1)INICIALIZAR"
1070 PRINT '"2)ENTRADA DE NUEVOS
 ELEMENTOS"
1080 PRINT '"3)BUSQUEDA/ELIMINAC
ION"
1090 PRINT '"5)GENERAR PREGUNTAS
"
1100 PRINT '"6)VER O PONER A CER
O PUNTUACION"
1110 PRINT '"7)FINALIZAR"
1120 INPUT "Cual requiere?";Z$
1130 CLS
1140 IF Z$="1" THEN GO SUB 1260
1150 IF Z$="2" THEN GO SUB 1390
1160 IF Z$="3" THEN GO SUB 1680
1170 IF Z$="5" THEN GO SUB 2010
1180 IF Z$="6" THEN GO SUB 2340
1190 IF Z$="7" THEN GO TO 1220
1200 CLS
1210 GO TO 1000
1220 PRINT AT 7,12;"PALABRAS"
1230 INPUT "Ha entrado alguna in
formacion  nueva que desee guar
dar? (S/N)";Q$
1240 IF Q$="S" THEN SAVE "PALABR
AS": PRINT AT 10,0;"REBOBINE, PU
LSE LUEGO UNA TECLA PARA VERIFIC
AR.": PAUSE 0: VERIFY "PALABRAS"
1250 STOP
```

Este es un módulo estándar de menú.

MODULO 4.2.2

```
1260>REM ***********************
1270 REM INICIALIZAR
1280 REM ***********************
1290 DIM O$(32)
1300 LET L$="[redacted]"
1310 LET ELEMENTOS=0
1320 LET I$="     "
1330 LET ELEMFIN=1
1340 LET BIEN=0
1350 LET QTOT=0
1360 DIM A$(50,15): DIM B$(50,40
0): DIM C$(15)
1370 DIM B(100)
1380 RETURN
```

Este módulo contiene las variables del programa. Las tablas se utilizan para guardar la cadena de caracteres compactada y la palabra asociada.

MODULO 4.2.3

```
2620>REM ***********************
2630 REM MOSTRAR TEXTOS
         EMPAQUETADOS
2640 REM ***********************
2650 BORDER 1: PAPER 7: CLS : PR
INT AT 1,1;: LET PANTALLA=0: LET
 CONTADOR=1
2660 IF P$(CONTADOR)=CHR$ 255 TH
EN LET CONTADOR=CONTADOR+1: IF P
$(CONTADOR)<>CHR$ 0 THEN PRINT O
$( TO CODE P$(CONTADOR));: LET P
ANTALLA=PANTALLA+CODE P$(CONTADO
R): PRINT " " AND PANTALLA/30=I
NT (PANTALLA/30);: LET CONTADOR=
CONTADOR+1: IF CONTADOR<=LEN P$
THEN GO TO 2660
2670 IF CONTADOR>LEN P$ THEN GO
TO 2710
2680 IF P$(CONTADOR)=CHR$ 0 THEN
 LET CONTADOR=CONTADOR+1
2690 LET PANTALLA=PANTALLA+1: IN
VERSE CODE P$(CONTADOR+2): PRINT
 P$(CONTADOR);: POKE (23296-32*(
PEEK 23689)-PEEK 23688+32),CODE
P$(CONTADOR+1): INVERSE 0: IF PA
NTALLA/30=INT (PANTALLA/30) THEN
 PRINT " ";
2700 LET CONTADOR=CONTADOR+3: IF
 CONTADOR<=LEN P$ THEN GO TO 266
0
2710 PAPER 7: INK 1: RETURN
```

Este módulo se ha sacado directamente del programa Artista y vuelve a visualizar la cadena compactada.

MODULO 4.2.4

```
1560>REM ************************
1570 REM INSERCION
1580 REM ************************
1590 IF LEN I$=4 THEN GO TO 1630
1600 LET LUGAR=256*CODE I$(3)+CO
DE I$(4)
1610 LET I$=I$( TO 2)+I$(5 TO )
1620 GO TO 1650
1630 LET LUGAR=ELEMFIN
1640 LET ELEMFIN=ELEMFIN+1
1650 LET A$(LUGAR)=W$
1660 LET B$(LUGAR)=P$: LET B(LUG
AR)=LEN P$
1670 RETURN
```

Este módulo tiene la misma estructura que el de inserción del programa anterior, excepto en que la variable que guarda los huecos de la tabla se llama I$ y que la dirección del nuevo registro no se guarda en una tabla de punteros. Teniendo tan sólo 5Ø registros parece innecesario el tener una compleja función de búsqueda para establecer el orden correcto de los registros.

MODULO 4.2.5

```
1390>REM ************************
1400 REM ENTRADA DE NUEVOS
         ELEMENTOS
1410 REM ************************
1420 PRINT "          NUEVOS ELEME
NTOS"
1430 INPUT "Nombre del dibujo a
cargar desde cinta? (ZZZ PARA SA
LIR)";N$
1440 PRINT ''"PULSE CUALQUIER TE
CLA Y ARRANQUE LA CINTA.": PAUSE
 0: PRINT ''"EFECTUANDOSE LA CAR
GA DE": LOAD N$ DATA P$()
1450 GO SUB 2620
1460 INPUT "Quiere este dibujo?(
S/N)";Q$
1470 IF Q$<>"S" THEN RETURN
1480 INPUT "Palabra asociada a e
ste dibujo?";W$
1490 PRINT AT 21,0;W$
1500 INPUT "Es correcto? (S/N)";
Q$
1510 CLS : IF Q$="N" THEN GO TO
1480
1520 GO SUB 1560: LET ELEMENTOS=
ELEMENTOS+1
1530 INPUT "Quiere entrar otro d
ibujo? (S/N)";Q$
1540 IF Q$="N" THEN RETURN
1550 GO TO 1420
```

Este módulo sirve para entrar el texto compactado que estaba guardado en cinta y que contiene el dibujo, lo visualiza, acepta la palabra que le corresponde y finalmente la guarda.

Comprobación del módulo 4.2.5

Habiendo entrado varios módulos que ya le parecerán familiares, sin comprobar su funcionamiento, ahora está en situación de comprobar que el programa entrará las tablas guardadas en la cinta y que han sido creadas y guardadas utilizando el programa Artista, visualizar el dibujo que contienen y guardarlo todo en memoria.

MODULO 4.2.6

```
1680 >REM ************************
1690 REM BUSQUEDA DE USUARIO
1700 REM ************************
1710 IF ELEMENTOS=0 THEN RETURN
1720 LET S=1
1730 IF B$(S,1)=" " AND S<50 THE
N LET S=S+1: GO TO 1730
1740 IF S=50 AND B$(S,1)=" " THE
N RETURN
1750 CLS : DIM P$(B(S)): LET P$=
B$(S,1 TO B(S)): GO SUB 2620: PR
INT AT 21,0;A$(S)
1760 PRINT AT 0,0;"             B
USQUEDA"
1770 PRINT '"ORDENESS DISPONIBLE
S:"
1780 PRINT '">""ENTER"" VER SIG.
 ELEMENTO"
1790 PRINT ">NUMERO POSITIVO O N
EGATIVO PARA MOVER PUNTERO"
1800 PRINT ">""DDD"" ELIMINAR EL
EMENTO"
1810 PRINT ">""ZZZ"" ABANDONAR F
UNCION"
1820 PRINT 'L$
1830 INPUT S$
1840 IF S$="DDD" THEN GO SUB 193
0: RETURN
1850 IF S$="ZZZ" THEN RETURN
1860 IF S$<>"" THEN GO TO 1890
1870 LET S=S+1
1880 GO TO 1730
1890 LET S=S+VAL S$
1900 IF S>=50 THEN RETURN
1910 IF S<1 THEN LET S=1
1920 GO TO 1730
```

Como antes, ésta es una versión simplificada del módulo de búsqueda, que se ha tomado del programa anterior. Cuando se visualiza un registro determinado no busca como antes primeramente la dirección en la tabla de punteros, sino que empieza sencillamente en la línea 1 de la tabla principal y visualiza lo que hay allí, a menos que el primer carácter sea un espacio, en cuyo caso la línea está vacía y el módulo pasa a la línea siguiente.

Comprobación del módulo 4.2.6

Ahora podrá recorrer y visualizar los registros guardados, avanzando hacia adelante o hacia atrás.

MODULO 4.2.7

```
1930>REM ************************
1940 REM ELIMINACIONES
1950 REM ************************
1960 LET A$(S)=""
1970 LET B$(S)=""
1980 LET ELEMENTOS=ELEMENTOS-1
1990 LET I$=I$( TO 2)+CHR$ INT (
S/256)+CHR$ INT (S-256*INT (S/25
6))+I$(3 TO )
2000 RETURN
```

Este módulo permite al usuario borrar registros y es una versión simplificada del módulo equivalente del programa anterior.

Comprobación del módulo 4.2.7

Ahora ya podrá borrar los registros que desee.

MODULO 4.2.8

```
2010>REM ************************
2020 REM PREGUNTAS ALEATORIAS
2030 REM ************************
2040 LET PREGUNTA=0
2050 DIM Q(5)
2060 LET Q1=INT (RND*ELEMENTOS+1
)
2070 LET Q2=INT (RND*5+1)
2080 LET Q(Q2)=Q1
2090 FOR I=1 TO 5
2100 IF I=Q2 THEN GO TO 2170
2110 LET S=INT (RND*ELEMENTOS+1)
2120 IF S=Q(Q2) THEN GO TO 2110
2130 FOR J=1 TO I
2140 IF S=Q(J) THEN GO TO 2110
2150 NEXT J
2160 LET Q(I)=S
2170 NEXT I
2180 DIM P$(B(Q1)): LET P$=B$(Q1
): GO SUB 2620
2190 PRINT AT 0,0;
2200 FOR I=1 TO 5: DIM T$(32): L
ET T$=A$(Q(I))
2210 PRINT INK 9; PAPER 7-I;T$
2220 NEXT I
2230 PRINT '"CUAL ES?"
2240 LET QTOT=QTOT+1
2250 INPUT C$: PRINT C$: IF C$=A
$(Q1) THEN GO TO 2270
```

```
2260 PRINT '"MAL ";A$(Q1): GO TO
 2300
2270 FOR I=0 TO 10: LET I1=I-8*(
I>7): PRINT AT I,0; OVER 1; PAPE
R I1;O$: NEXT I
2280 FLASH 1: PRINT AT Q2-1,13;A
$(Q1): PRINT AT 10,10;"BIEN!": P
AUSE 200: FLASH 0
2290 LET BIEN=BIEN+1
2300 INPUT """ENTER"" para nueva
 pregunta o ""ZZZ"" para termina
r";Q$
2310 CLS
2320 IF Q$="ZZZ" OR Q$="zzz" THE
N GO TO 2520
2330 GO TO 2050
```

Este módulo genera aleatoriamente las preguntas y las posibles respuestas. Es menos complejo que el módulo equivalente del programa anterior ya que no están previstos los tipos de registros.

Comentario

Líneas 22ØØ-228Ø. Obsérvese como es posible conseguir que la presentación de las preguntas y la recompensa para una respuesta correcta, sean atractivas visualmente sin un gran esfuerzo. En estas líneas las variables del bucle se utilizan para seleccionar los colores de fondo y añadir un cierto interés a la pantalla.

Comprobación del módulo 4.2.8

Ahora el programa debe ser capaz de generar preguntas.

MODULO 4.2.9

```
2340>REM ***********************
2350 REM PUNTUACION
2360 REM ***********************
2370 PRINT "             PUNTUACI
ON"
2380 IF QTOT<>0 THEN GO TO 2430
2390 PRINT ''''"       AUN NO HAY
 PUNTUACION."
2400 PRINT '"Cualquier tecla par
a continuar."
2410 PAUSE 0
2420 RETURN
2430 PRINT '"TOTAL DE PREGUNTAS:
";QTOT
2440 PRINT '"CORRECTAS:";BIEN
2450 DEF FN A()=INT (((BIEN-QTOT
/5)/(QTOT*.8))*100): PRINT '"EVA
LUACION:";FN A();" POR CIENTO."
2460 PRINT '"DESEA PONERLA A CER
```

```
O? (S/N)"
2470 INPUT Q$
2480 IF Q$<>"S" THEN RETURN
2490 LET QTOT=0
2500 LET BIEN=0
2510 RETURN
```

Este es el módulo de puntuación que se ha tomado del programa anterior, con una pequeña modificación.

Comentario

Línea 2450. La evaluación se efectúa mediante la función FN A ().

Comprobación del módulo 4.2.9

Ahora podrá pedir la puntuación desde el menú.

MODULO 4.2.10

```
2520>REM ***********************
2530 REM ADIOS
2540 REM ***********************
2550 CLS : LET I1=0: LET I2=1: P
APER 9
2560 FOR I=1 TO 100
2570 LET I1=I1+1-25*(I1=24)
2580 LET I2=I2+1-8*(I2=7)
2590 POKE 23692,255: INK I2: PRI
NT 'TAB I1;"ADIOS!";
2600 NEXT I: PRINT " LA PUNTUACI
ON FUE ";FN A()
2610 INK 0: PAPER 7: STOP
```

Este módulo visualiza una despedida del programa, ligeramente menos aburrida, y utiliza la función definida en el módulo de puntuación para dar una evaluación final.

Comentario

Línea 259∅. Obsérvese la utilización de este POKE, que se utiliza para evitar el desplazamiento de las líneas y que se detiene una vez visualizadas 22 líneas. La dirección de memoria 23692 es utilizada por el Spectrum para guardar el número de líneas que deben permanecer visualizadas antes de que se pida al usuario si debe continuar el «scrolling» o desplazamiento de las líneas hacia arriba.

Comprobación del módulo 4.2.10

Este módulo tan sólo puede llamarse desde el módulo que genera las preguntas.

Resumen

Este, como antes, es un programa que requiere la realización previa de cierto trabajo si se quiere que sea de cierta utilidad, ya que los pequeños dibujos que utiliza necesitarán cierto tiempo para ser creados. En la creación de estos dibujos es especialmente importante el prepararlos adecuadamente con anterioridad antes de sentarse a trabajar con el programa Artista. La utilización del papel cuadriculado puede ahorrar una gran cantidad de frustraciones cuando se intente crear los dibujos antes de entrarlos. Utilice papel cuadriculado para poder contar el número de caracteres del dibujo. Cualquier dibujo que tenga más de 1ØØ o cuyos caracteres estén muy separados en la pantalla es muy probable que no quepa en el espacio de 4ØØ caracteres reservado para el texto compactado. Probablemente habrá observado que la parte del programa que administra el test está más bien separada de las otras funciones. Esto se hace así para que la persona adulta pueda empezar el test y dejarlo en manos de los niños sabiendo que es muy poco probable que puedan reentrar el programa principal y crear problemas o hacer trampas. Antes de empezar el test probablemente sea mejor pulsar la tecla CAPS LOCK para que seleccione las letras minúsculas, ya que los niños tienen tendencia a estar más familiarizados con las letras de este tipo. Para poder hacer esto, las palabras asociadas con cada dibujo también deben haberse entrado en minúsculas.

Si no se tiene a mano ningún niño de la edad adecuada con el que practicar, entonces ¿por qué no diseñar algunas sencillas representaciones de símbolos eléctricos, y hacer que el programa genere algunos tests para identificar circuitos electrónicos sencillos. Como dije antes, si se tiene algo que funciona, adáptelo.

Posibles mejoras

1) Si realmente no se necesita un registro de la característica de inversión para cada carácter dentro del texto compactado, se podría o bien aumentar el número de dibujos que el programa puede guardar o aumentar su complejidad.
2) Este es otro caso en que la cooperación con otros usuarios del Spectrum para intercambiar dibujos puede ahorrarle muchos esfuerzos.

4.3 Geografía

Este es un programa poco complicado que comprueba de forma efectiva sus conocimientos sobre geografía, o al menos la situación de las ciudades de una variedad de países.

Para utilizar el programa tendrá que modificar el programa Artista dado anteriormente. La modificación consiste en añadir un módulo que coge la retícula de 2Ø*3 contenida en A$ (1) y la guarda en una cinta. Esta modificación se deja para que la realice usted y probablemente elegirá o bien realizar una versión de Artista que únicamente cree y trabaje con una tabla A$(2Ø,3Ø) y que no guarde las características de color o cree un texto compactado, o bien añadir al programa Artista existente un sencillo bucle como FOR I=1 TO 2Ø:LET B$(I)=A$(1,I):NEXT I, transfiera el contenido de A$(1) a otra tabla.

En cualquier caso deberá obtener un programa que sea capaz de guardar un texto de 2Ø*3Ø, que contenga los caracteres de una pantalla. El siguiente paso es crear algunos dibujos que reproduzcan el perfil de un país y que lo guarde con el nombre del país que representa. Es entonces cuando podemos coger el hilo de este programa.

MODULO 4.3.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 PRINT "          GEOGRAFIA"
1040 PRINT '"1)CARGAR NUEVO PAIS
"
1050 PRINT '"2)REGISTRAR NUEVAS
CIUDADES"
1060 PRINT '"3)PREGUNTAS"
1070 PRINT '"4)INICIALIZAR"
1080 PRINT '"5)FINALIZAR"
1090 INPUT Z$: CLS : LET QTOTAL=
0: LET MAL=0
1100 IF Z$="1" THEN GO SUB 1260
1110 IF Z$="2" THEN GO SUB 1320
1120 IF Z$="3" THEN LET PAIS=FN
A(): GO SUB 1670: GO SUB 1510
1130 IF Z$="5" THEN GO TO 1160
1140 IF Z$="4" THEN GO SUB 1190
1150 CLS : GO TO 1000
1160 PRINT AT 10,10;"GEOGRAFIA"
1170 INPUT "Ha entrado nuevos da
tos que dese guardar? (S/N)";Q$:
 IF Q$="S" THEN SAVE "GEOGRAFIA"
: PRINT "Rebobine, luego ""ENTER
"" para   verificar.": PAUSE 0:
 VERIFY "GEOGRAFIA"
1180 STOP
```

Este es un módulo estándar de menú.

MODULO 4.3.2

```
1190>REM ************************
1200 REM INICIALIZAR
1210 REM ************************
1220 DIM B$(10,20,30)
1230 LET PAISES=0
1240 DEF FN A()=INT (RND*PAISES+
1)
1250 RETURN
```

Las variables se inicializan aquí.

Comentario

Línea 122Ø. Los textos que contienen los países se guardan en la tabla B$.

Línea 124Ø. Esta función elige aleatoriamente un país.

MODULO 4.3.3

```
1260>REM ************************
1270 REM CARGAR NUEVO PAIS
1280 REM ************************
1290 INPUT "NOMBRE DEL PAIS A CA
RGAR? ";N$: INPUT "ARRANCAR CINT
A, LUEGO ""ENTER""";Q$: LOAD N$
DATA A$()
1300 INPUT "NUMERO DE POSICION:"
;LUGAR
1310 LET PAISES=PAISES+1: FOR I=
1 TO 20: LET B$(LUGAR,I)=A$(I):
NEXT I: RETURN
```

Este módulo acepta la entrada desde la cinta de una tabla de 2Ø×3Ø con el nombre correspondiente del país y lo coloca en B$, en la posición decidida por el usuario.

Comprobación del módulo 4.3.3

Cargue un texto y colóquelo en B$: compruebe en modo directo que se ha guardado adecuadamente.

MODULO 4.3.4

```
1320>REM ************************
1330 REM REGISTRAR CIUDADES
1340 REM ************************
1350 INPUT "NUMERO DE PAIS? ";PA
IS
```

```
1360 PRINT : FOR I=1 TO 20: PRIN
T PAPER 5; INK 2;TAB 1;B$(PAIS,I
): NEXT I: LET X=1: LET Y=1: LET
 T$="": GO TO 1430
1370 PRINT OVER 1; PAPER 8; INK
0;AT X,Y;"*": PAUSE 2: PRINT OVE
R 1; PAPER 8; INK 0;AT X,Y;"*"
1380 LET T$=INKEY$: IF T$="" THE
N GO TO 1370
1390 IF T$<"5" OR T$>"9" THEN GO
 TO 1370
1400 PRINT PAPER 5; INK 2;AT X,Y
;B$(PAIS,X,Y)
1410 LET X=X+(T$="6")-(T$="7"):
LET X=X+(X<1)-(X>20)
1420 LET Y=Y+(T$="8")-(T$="5"):
LET Y=Y+(Y<1)-(Y>30)
1430 PRINT INK 0;AT 0,0;"X=";X;"
  Y=";Y;"   ""9"" PARAR": IF T$<>
"9" THEN GO TO 1370
1440 PRINT AT 3,0;"LAS SENTENCIA
S DATA TENDRAN EL  SIGUIENTE ASP
ECTO:"'5000+PAIS*100;" DATA n,""
ELPAIS"""'5000+PAIS*100+1;" DATA
 ""LACIUDAD"",6,12"
1450 PRINT '"PONER NOMBRE DE CIU
DAD, X & Y EN LA SENTENCIAS DATA
 SIGUIENTES A ";5000+PAIS*100;",
";
1460 PRINT "INCREMENTANDO EL NUM
ERO DE SENTENCIA DATA EN 1 POR C
ADA    NUEVA CIUDAD."
1470 PRINT '"EL PRIMER ELEMENTO
DE LA LINEA    ";5000+PAIS*100;" D
EBE SER EL TOTAL DE       CIUDAD
ES, EL SEGUNDO, EL NOMBRE DEL PA
IS."'"PULSAR UNA TECLA PARA LIST
AR SENTENCIAS DATA EXISTENTES."
1480 PAUSE 0: LIST 5000+PAIS*100
: STOP
```

Este módulo permite al usuario mover un cursor sobre el dibujo hasta que queden establecidas las coordenadas de una ciudad a satisfacción del usuario. Entonces estas coordenadas podrán entrarse en una sentencia de DATA al final del programa.

Comentario

Línea 136Ø. La sección correspondiente de la tabla B$ se visualiza en la pantalla.

Líneas 137Ø-142Ø. Estas son las rutinas estándar para el movimiento del cursor sobre la pantalla.

Línea 143Ø. Las coordenadas del cursor se visualizan en la pantalla.

Líneas 144Ø-148Ø. Si el usuario detiene el programa, se dan instrucciones para la entrada de las coordenadas en las sentencias DATA, al final del programa. La estructura de la sentencia de DATA es la siguiente:

1) El nombre de cada país se guarda en una sentencia de DATA cuyo número de línea será 5ØØØ+(1ØØ veces su número en B$). El nombre del país va precedido, en esta línea de DATA, por el número de sus ciudades cuyas coordenadas van a registrarse.
2) Los datos correspondientes a las ciudades se guardan en la línea que va inmediatamente a continuación, es decir 51Ø1, 51Ø2 en la forma de líneas sencillas, cada una conteniendo el nombre de la ciudad, seguido por las coordenadas X e Y.

Si la ciudad es la primera en entrarse para un país que acaba de cargarse, el país deberá guardarse en el lugar indicado por el programa. Si no es la primera ciudad, el elemento que registra el número de ciudades deberá incrementarse y la nueva ciudad deberá entrarse en una línea propia. El programa da al usuario el número correcto de líneas donde hay que entrar los datos y dos líneas de ejemplo. Este es un método extremadamente sencillo a la hora de utilizarlo para entrar nuevos elementos y sirve como otro ejemplo del ahorro que puede conseguirse en la complejidad de los programas si el usuario puede realizar un poco de precompactación de la información con la que tiene que trabajar el programa. Si no hubiésemos utilizado líneas de DATA tendríamos que haber previsto una tabla que guardase los nombres de las ciudades y las coordenadas, y además un método para clasificar y otro para borrar.

Comprobación del módulo 4.3.4

Ahora ya podrá mover el cursor sobre la pantalla y detener el programa con la entrada de un "9", tras lo cual deberán aparecer las instrucciones para la entrada de las coordenadas de la ciudad en las sentencias de DATA.

MODULO 4.3.5

```
1670>RESTORE (5000+100*PAIS): RE
AD N,C$: LET CIUDAD=INT (RND*N+1
): RESTORE (5000+100*PAIS+CIUDAD
): READ M$,X,Y: RETURN
```

Esta línea lee los datos para un país y una ciudad, indicados por las variables PAIS y CIUDAD. Es un buen ejemplo de la flexibilidad de la función RESTORE en el Spectrum.

```
1510>REM ***********************
1520 REM CUAL CIUDAD?
1530 REM ***********************
1540 CLS : PRINT INK 0;C$: FOR I
=1 TO 20: PRINT PAPER 5; INK 2;T
AB 1;B$(PAIS,I): NEXT I
1550 PRINT OVER 1; PAPER 8; INK
0; FLASH 1;AT X,Y;"*"
1560 FOR I=1 TO 3
1570 LET QTOTAL=QTOTAL+1: INPUT
"NOMBRE DE ESTA CIUDAD? ";N$: IF
 N$=M$ THEN GO TO 1620
1580 PRINT AT 0,0;"MAL!
                  ": PAUSE 100: P
RINT AT 0,0;C$;" ": LET MAL=MAL+
1: NEXT I
1590 PRINT AT 0,0;"LA CIUDAD ERA
:";M$: INPUT """ENTER"" PARA CON
TINUAR";Q$
1600 PRINT INK 2; PAPER 5; FLASH
 0;AT X,Y;B$(PAIS,X,Y)
1610 GO TO 1630
1620 PRINT AT 0,0;"";M$;"
 ": FOR I=1 TO 20: PRINT FLASH 1
; INK 2; PAPER 5;AT I,1;B$(PAIS,
I): PRINT INK 1;AT I,I;"BIEN!":
NEXT I: PRINT OVER 1; FLASH 1;AT
 X,Y;"*"
1630 PRINT INK 0;AT 21,0;"PREGUN
TAS:";QTOTAL;" BIEN:";QTOTAL-MAL
;" %:";(QTOTAL-MAL)/QTOTAL*100
1640 INPUT "OTRA VEZ? (S/N)? ";Q
$: IF Q$="N" THEN RETURN
1650 INPUT "EL MISMO PAIS? (S/N)
? ";Q$: IF Q$="N" THEN LET PAIS=
FN A()
1660 GO SUB 1670: GO TO 1510
```

Este módulo visualiza un país, añade un indicador en la posición de una ciudad y pide la entrada del nombre de esta ciudad. Basándose en las respuestas dadas hasta el momento, se visualiza una evaluación de la actuación del usuario hasta el momento.

Comentario

Línea 157∅. El nombre entrado se compara con el nombre sacado de la sentencia DATA. Obsérvese que se permiten tres intentos para cada ciudad.

Línea 162∅. Una sencilla recompensa visual se da si el usuario entra el nombre correcto.

Línea 165∅. Observe la utilización de la función para determinar un nuevo país si el usuario desea cambiar los países.

Comprobación del módulo 4.3.6

El programa deberá ahora ser capaz de plantear un test basado en los datos entrados anteriormente.

Resumen

Se puede llegar a un abuso de las sentencias de DATA. Mucha gente lo hace utilizándolas en programas donde los datos deben cambiarse frecuentemente o clasificarse de una forma u otra. Sin embargo, utilizándolas adecuadamente, las sentencias DATA pueden ser, además de convenientes, un ahorro de tiempo, ya que los datos pueden añadirse o borrarse sin la constante modificación de grandes tablas. Puede que ésta no sea una regla a seguir siempre, pero si un programa se detiene continuamente para pedir al usuario que entre nuevos datos entonces es que el programador tal vez está evitando la tarea de crear un programa que pueda tratar con el tipo de datos necesarios en la forma que realmente necesita.

Posibles mejoras

1) Podría ampliar el programa de forma que pudiera visualizar, ríos y montañas y otras características físicas y pedir el nombre. No es fácil, pero es posible e impresionaría mucho.
2) ¿Podría pensar en otras aplicaciones de este programa? Una que se me ocurre es plantear preguntas sobre el análisis de circuitos para aquellos que aprenden electrónica, por ejemplo, ¿cuál es la corriente en el punto indicado por el cursor parpadeante?

5. Programas de utilidad. Una colección de rutinas variadas

En este capítulo consideramos seis programas muy distintos bajo el titulo de «programas de utilidad» que ilustran como mínimo la gran variedad de aplicaciones que puede elegir para sus programas, para responder a sus propias necesidades particulares. Los seis programas son:

1) Calculadora, que está diseñado para conseguir que el Spectrum sea todavía más útil cuando se trate de series de cálculos repetitivos.
2) Calorías, un programa que quizás ilustra las propias preocupaciones del autor, pero que no obstante es una herramienta conveniente para aquellos que necesiten vigilar su peso.
3) Gráficas, un programa de propósito general para el trazado de gráficas.
4) Renumeración, una herramienta vital para sus propios programas, que le permitirá pulir aquellos programas numerados de forma irregular.
5) Archivo II. Volvemos a considerar el programa Archivo y lo desarrollamos de forma que pueda aceptar datos de tamaño y estructura irregular. Ahora se convierte en un programa mucho más potente.
6) Mecanografía, un sencillo programa que puede enseñarle a teclear mejor.

5.1. Calculadora

Quizá le parezca extraño el diseñar un programa para hacer más fáciles los cálculos en un ordenador. Después de todo, ¿no es el Spectrum capaz de aceptar fórmulas en modo directo para cálculos directos o en forma de programa para cálculos regulares?

La respuesta es naturalmente que el Spectrum puede hacer las dos cosas. Pero entre los cálculos directos que pueden entrarse fácilmente y el uso especializado que se realiza con regularidad suficiente para justificar la escritura de un programa especial, como en el caso de los programas financieros del capítulo 2, existe un amplio

campo de necesidades en el que deben realizarse cálculos repetitivos y que sería demasiado pesado entrarlos en modo directo y, no obstante, no son lo suficientemente importantes para que valga la pena el hacer un programa especial para ellos. Para tales aplicaciones, necesitamos un programa que permita entrar una gran variedad de fórmulas y variables, que puedan modificarse fácilmente y visualizar sus resultados.

Una vez empezado un proyecto de este tipo quizá también podríamos, al mismo tiempo, diseñar un programa en el que los resultados de cambiar el valor de las variables pudieran compararse en la forma de un programa de «Qué sucedería si», como el programa de presupuestos del capítulo 2.

Para conseguir esto sin tener que escribir constantemente nuevas fórmulas en las líneas del programa, utilizaremos dos de las características más atractivas del BASIC de Sinclair: su flexible manejo de textos y su habilidad para evaluar expresiones numéricas guardadas en un texto. Utilizándolas conjuntamente, estas dos características nos permitirán realizar tareas que serían casi imposibles en muchas otras máquinas cuyo precio excede con mucho el del Spectrum.

MODULO 5.1.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : INK 0: PAPER 7: PRINT
"                 CALCULADORA"
1040 PRINT '"1)INICIALIZAR"
1050 PRINT '"2)VISUALIZAR TABLA"
1060 PRINT '"3)CALCULAR TABLA"
1070 PRINT '"4)DEFINIR VARIABLES
"
1080 PRINT '"5)DEFINIR LINEAS"
1090 PRINT '"6)VISUALIZAR COLUMN
AS            DE VARIABLES"
1100 PRINT '"7)FINALIZAR"
1110 INPUT Z$: CLS
1120 IF Z$="1" THEN GO SUB 1220
1130 IF Z$="2" THEN GO SUB 1300
1140 IF Z$="3" THEN GO SUB 1400
1150 IF Z$="4" THEN GO SUB 1690
1160 IF Z$="5" THEN GO SUB 1810
1170 IF Z$="6" THEN GO SUB 1920
1180 IF Z$="7" THEN GO TO 1200
1190 CLS : GO TO 1030
1200 PRINT AT 10,10;"CALCULADORA
": INPUT "HA ENTRADO NUEVOS DATO
S QUE       QUIERA GUARDAR? (S/N)
";Q$: IF Q$="S" THEN SAVE "CALCU
LADOR": BEEP 1,40: PRINT "REBOBI
NE Y PULSE UNA TECLA PARA VERIFI
CAR.": PAUSE 0: VERIFY "CALCULAD
OR": PRINT "VERIFICADO"
1210 STOP
```

Otro módulo estándar de menú.

MODULO 5.1.2

```
1220 REM ***********************
1230 REM INICIALIZAR
1240 REM ***********************
1250 DIM A$(20,10): DIM B$(20,10
): DIM C$(20,50): DIM E$(20,50)
1260 DIM B(20,10): DIM D(20,10)
1270 LET LINEAS=0: LET VARIABLES
=0
1290 RETURN
```

La utilización de las distintas tablas se explicará en el transcurso del programa.

MODULO 5.1.3

```
1690 REM ***********************
1700 REM DEFINIR VARIABLES
1710 REM ***********************
1720 CLS : FOR I=1 TO VARIABLES:
 PRINT I;") ";B$(I): NEXT I
1730 INPUT "1)EXISTENTE   2)NUEV
A   3)SALIR ";Q$: IF Q$="3" THEN
 RETURN
1740 IF Q$="1" THEN INPUT "NUMER
O? ";N: GO TO 1760
1750 IF Q$="2" THEN INPUT "NOMBR
E DE LA VARIABLE? ";N$: LET VARI
ABLES=VARIABLES +1: LET N=VARIAB
LES: LET B$(VARIABLES)=N$
1760 CLS : PRINT B$(N): FOR I=1
TO 10: PRINT I;") ";B(N,I): NEXT
 I
1770 INPUT "NUMERO█""ZZZ"" SALIR
█""DDD"" BORRAR ";Q$: IF Q$="ZZ
Z" THEN GO TO 1720
1780 IF Q$="DDD" THEN FOR I=N TO
 VARIABLES-1: LET B$(I)=B$(I+1):
 FOR J=1 TO 10: LET B(I,J)=B(I+1
,J): NEXT J: NEXT I: LET B$(VARI
ABLES)="": FOR I=1 TO 10: LET B(
VARIABLES,I)=0: NEXT I: LET VARI
ABLES=VARIABLES-1: RETURN
1790 INPUT "VALOR? (""X"" IGUAL
COLUMNA UNO) ";N$: IF N$="X" THE
N LET B(N,VAL Q$)=B(N,1): GO TO
1760
1800 LET B(N,VAL Q$)=VAL N$: GO
TO 1760
```

El propósito de este módulo es permitir al usuario nombrar 2Ø variables distintas y especificar una serie de 1Ø valores para cada variable nombrada, permitiendo así realizar fácilmente cálculos comparativos.

Comentario

Líneas 172∅-173∅. Los nombres de las variables definidas hasta el momento están guardadas en la tabla B$ y se visualizan al entrar en este módulo.

Línea 175∅. Los nombres de las nuevas variables se colocan en la tabla en una posición definida por la variable VARIABLES, que se pone a cero al empezar y se incrementa para cada nuevo nombre.

Línea 176∅. Los 1∅ valores asociados con la variable especificada se guardan en la tabla B. Estos valores se visualizan para que el usuario pueda especificar cambios.

Comprobación del módulo 5.1.3

Las variables especificadas no tienen por ahora una utilización práctica, pero el módulo debe ser capaz de aceptar 2∅ nombres, junto con 1∅ valores asociados y borrar nombres y valores.

MODULO 5.1.4

```
1920>REM ***********************
1930 REM VISUALIZAR COLUMNA
1940 REM ***********************
1950 CLS : INPUT "QUE COLUMNA? (
0 PARA SALIR) ";N: IF N=0 THEN R
ETURN
1960 CLS : PRINT "COLUMNA ";N
1970 FOR I=1 TO VARIABLES: PRINT
 I;") ";B$(I);" ";B(I,N): NEXT I
1980 INPUT "NUMERO PARA CORRECCI
ON█ ""ZZZ""  SALIR";Q$: IF Q$="
ZZZ" THEN GO TO 1950
1990 INPUT "NUEVO VALOR DE LA VA
RIABLE? ";R: LET B(VAL Q$,N)=R:
GO TO 1960
```

El módulo anterior permitía la visualización de todos los valores asociados a cada variable. Este módulo visualiza el primero, el segundo o enésimo valor asociado con cada variable. Así, si deben realizarse una serie de cálculos sobre una serie de datos anuales, este módulo visualizará el valor de cada variable para un año determinado.

Comprobación del módulo 5.1.4

Si ha entrado los nombres de algunas variables y algunos valores asociados, podrá visualizar una columna de ellos.

```
181Ø >REM ************************
182Ø  REM DEFINIR LINEAS
183Ø  REM ************************
184Ø  CLS : PRINT "               LI
NEAS"
185Ø  FOR I=1 TO LINEAS: PRINT I;
")";"" +A$(I)+ "";C$(I): NEXT I
186Ø  INPUT "1)EXISTENTE    2)NUEV
A   3)BORRAR   4)SALIR";Q$: IF Q
$="4" THEN RETURN
187Ø  IF Q$="2" THEN INPUT "TITUL
O DE LINEA? ";N$: LET LINEAS=LIN
EAS+1: LET N=LINEAS: LET A$(LINE
AS)=N$: INPUT "FORMULA? ";F$: LE
T C$(LINEAS)=F$: GO SUB 149Ø: LE
T E$(N)=G$: GO TO 184Ø
188Ø  IF Q$="1" THEN INPUT "NUMER
O? ";N: INPUT "TITULO NUEVO? (S/
N) ";T$: IF T$="S" THEN INPUT "E
SPECIFICAR NUEVO TITULO: ";T$: L
ET A$(N)=T$
189Ø  IF Q$="1" THEN INPUT "NUEVA
 FORMULA? (S/N) ";T$: IF T$="S"
THEN INPUT "ESPECIFICAR NUEVA FO
RMULA: ";F$: LET C$(N)=F$: GO SU
B 149Ø: LET E$(N)=G$: GO TO 184Ø
19ØØ  IF Q$="3" THEN INPUT "NUMER
O? ";N: FOR I=N TO LINEAS-1: LET
 A$(I)=A$(I+1): LET C$(I)=C$(I+1
): NEXT I: LET A$(LINEAS)="": LE
T C$(LINEAS)="": LET LINEAS=LINE
AS-1
191Ø  RETURN
```

Para comprender este módulo tendrá que saber un poco cómo funciona el Spectrum al evaluar una expresión guardada en un texto o variable alfanumérica. Esa parece a primera vista una característica poco importante. Para qué puede servir el poder decirle al Spectrum que visualice el valor de "1+1" (PRINT VAL "1+1") —o de cualquier otra expresión para el caso— cuando puede hacerse más fácilmente mediante PRINT 1+1. La respuesta es que mientras una expresión numérica directa como 1+1 tan sólo puede incorporarse en una línea de programa, "1+1" puede colocarse en una línea de programa o guardarse y manipularse al igual que cualquier otro texto. Este módulo utiliza este hecho para guardar las fórmulas entradas por el usuario.

Comentario

Línea 185Ø. El número de fórmulas entradas hasta el momento se guarda en la variable LINEAS. La fórmula, en la forma entrada por el usuario, se guarda en la tabla A$ que limita la longitud de cada una a 5Ø caracteres.

Línea 187Ø. La fórmula, que puede tener hasta 5Ø caracteres de

largo como indicamos anteriormente, puede ser cualquier cosa que el Spectrum pueda reconocer como algo que tenga un valor. La única limitación es que los nombres de variable no pueden contener números. Así AA sería reconocido como un nombre de variable, mientras que A1, no. La única excepción es que una fórmula puede utilizar los resultados de otra, incluyendo una variable tal como LINEA1+1Ø, que en este caso se evaluaría como el resultado de la primera fórmula entrada, más 1Ø.

Comprobación del módulo 5.1.5

Este módulo no puede comprobarse totalmente hasta que no se hayan entrado los dos siguientes. Si se entra temporalmente la línea 149Ø LET G$=F$: RETURN, el módulo deberá aceptar fórmulas, junto con títulos identificativos breves de hasta 1Ø caracteres, y deberá permitir cambios y borrados.

MODULO 5.1.6

```
1490>REM ************************
1500 REM IDENTIFICAR VARIABLES
1510 REM ************************
1520 LET G$="": LET V$=""
1530 FOR I=1 TO LEN F$: IF F$(I)
=" " THEN GO TO 1610
1540 IF F$(I)>="A" AND F$(I)<="Z
" THEN LET V$=V$+F$(I): GO TO 16
00
1550 IF LEN V$>=5 THEN IF V$(1 T
O 5)="LINEA" AND (F$(I)>="0" AND
 F$(I)<="9") THEN LET V$=V$+F$(I
): GO TO 1600
1560 IF LEN V$>=5 THEN IF V$(1 T
O 5)="LINEA" AND LEN V$=5 AND ((
F$(I)<"0" OR F$(I)>"Z") OR (F$(I
)<"A" AND F$(I)>"9")) THEN PRINT
 "LINEA NO DEFINIDA": PAUSE 200:
 RETURN
1570 IF LEN V$>=5 THEN IF V$(1 T
O 5)="LINEA" AND (F$(I)<"A" OR F
$(I)>"Z") THEN GO SUB 1630: LET
V$="": LET G$=G$+F$(I): GO TO 16
00
1580 IF V$<>"" AND (F$(I)<"A" OR
 F$(I)>"Z") THEN GO SUB 1630: LE
T G$=G$+F$(I): LET V$="": GO TO
1600
1590 LET G$=G$+F$(I)
1600 NEXT I
1610 IF LEN V$>0 THEN GO SUB 163
0
1620 RETURN
```

Una vez entrado algo que sea reconocible por el Spectrum como una fórmula, nos encontramos con un problema —el de dar un valor a las variables que contiene la fórmula—. Si la fórmula contuviese una

variable de nombre STOTAL, un nombre válido para el programa, no podría obtenerse el resultado de la fórmula a menos que el Spectrum tuviese un valor para esta variable, algo que tan sólo puede hacerse utilizando una sentencia INPUT o LET, por ejemplo, LET STOTAL=1Ø.

Sería difícil llamar a este procedimiento flexible y fácil de utilizar comparado con la forma en que se nombraron y se les dieron valores a las variables en el módulo 3. El hecho es que mientras que los elementos guardados en B$ pueden llamarse variables, desde el punto de vista del programa, no son nada parecido para el Spectrum y no serán reconocidos como tales si se incluyen en una fórmula que debe evaluarse. Los nombres de variables entrados por el usuario nunca se utilizan en los cálculos realizados por este programa. Son sustituidos por otros nombres de variables para los cuales el Spectrum sí que tiene un valor correspondiente.

En este módulo, y en el siguiente, la fórmula entrada por el usuario se traduce a una forma que puede ser manejada por el Spectrum. Ya se ha dicho que pueden entrarse hasta 1Ø valores junto con el nombre de la variable y que estos 1Ø valores se guardan en la tabla B. Por lo tanto, para el Spectrum, cada uno de estos valores tiene un nombre del tipo B (X,Y), donde X es el número del nombre de la variable asociado en B$ e Y es el número de la columna del 1 al 1Ø. Estos dos módulos crean una nueva fórmula para sustituir la entrada por el usuario, cuyas variables serán elementos tomados de la tabla B. Se puede pedir que el Spectrum evalúe esta segunda fórmula utilizando la función VAL, ya que todas sus variables están definidas. Así, una fórmula entrada por el usuario como CIRCULO* PI* RADIO^ 2 se convertiría en algo tal como B(1,3)*PI* $B(2,3)^2$.

Comentario

Línea 152Ø. G$ se utilizará para guardar la fórmula traducida, a medida que se va construyendo, y por último se guardará en la tabla E$ en la posición equivalente a la que ocupaba la fórmula inicial en A$. B$ se utiliza para guardar temporalmente los nombres identificados en la fórmula entrada por el usuario.

Línea 153Ø. La fórmula definida por el usuario se guardará en una línea de 5Ø caracteres de la tabla y por lo general no rellenará este espacio. El módulo considera terminada la fórmula cuando encuentra el primer espacio vacío.

Línea 154Ø. Si se encuentra un carácter en la fórmula definida por el usuario, comprendido entre la A y la Z, entonces se interpreta como parte del nombre de la variable y se añade a lo que ya esté contenido en V$.

Línea 155∅. La primera sentencia de esta línea quizá le parezca un poco extraña; de hecho es necesaria para evitar un mensaje de error si V$ tiene menos de cuatro caracteres de largo y se especifica una condición tal como IF V$ (1 TO 5)="LINEA". Si V$ tiene menos de cinco caracteres de largo, esta línea no se ejecuta y si V$ tiene cinco o más caracteres de largo la sentencia no representa ninguna diferencia para la ejecución del programa. Una utilización interesante de la forma en que se comporta el Spectrum cuando se encuentra una condición falsa. El propósito de la parte principal de la línea es el proporcionar la posibilidad de reconocer variables del tipo LINEA1, LINEA2, etc. que se mencionaron anteriormente.

Línea 156∅. Esta línea comprueba que si una variable empieza con las letras LINEA, tenga un número identificativo a continuación.

Línea 157∅. Esta línea reconoce cuándo una variable LINEA está completa, identificando el primer carácter que no puede estar contenido en esta variable.

Línea 158∅. Volviendo al tipo normal de nombre de variable, que tan sólo podrá contener letras, esta línea reconoce cuándo una variable de este tipo está completa.

Línea 159∅. Los caracteres que no forman parte de un nombre de variable se añaden a la fórmula traducida.

Línea 161∅. Generalmente, el final de un nombre de variable se reconoce por que:

a) V$ contiene algunos caracteres y
b) El siguiente carácter encontrado no puede formar parte de una variable.

Evidentemente esto no puede funcionar si el nombre de la variable está al final de la fórmula, por lo que esta línea se utiliza para comprobar si V$, que se vacía tras cada identificación infructuosa, contiene algo.

Comprobación del módulo 5.1.6

Esta comprobación tan sólo puede realizarse tras la entrada del próximo módulo.

MODULO 5.1.7

```
1630>REM ************************
1640 REM EVALUAR
1650 REM ************************
1660 IF LEN V$>=5 THEN IF V$(1 T
```

```
0 5)="LINEA" THEN LET G$=G$+"D("
+V$(6 TO )+",J)": RETURN
1670 FOR J=1 TO VARIABLES: IF V$
<>B$(J,1 TO LEN V$) THEN NEXT J:
 PRINT "VARIABLE ";V$;" NO HALLA
DA.": PAUSE 200: LET IND=0: RETU
RN
1680 LET G$=G$+"B("+STR$ J+",J)"
: RETURN
```

Una vez que el módulo anterior ha identificado los nombres de las variables en la fórmula definida por el usuario, este módulo traduce estos nombres a términos que pueda manejar el Spectrum en los cálculos.

Comentario

Línea 166Ø. Los resultados de las evaluaciones de las fórmulas se colocarán después en la tabla D. Esta línea convierte un nombre de variable, como LINEA3, en el nombre de un elemento de D, tal como D(3,J). «J» se utiliza aquí ya que en un punto posterior la fórmula guardada se evaluará mediante dos bucles, un bucle con la variable I que se referirá al número de fórmulas y un bucle con la variable J que se referirá a cada uno de los 1Ø posibles valores para cada variable.

Línea 167Ø. En el caso de una variable ordinaria, esta línea comprueba los nombres de variables existentes e informa al usuario si esta variable no ha sido definida.

Línea 168Ø. Si el nombre de la variable ya ha sido declarado, el elemento correspondiente de la tabla B se añade a G$. Si la variable STOTAL fuera la tercera en la lista de variables, se convertiría en B(3,J).

Comprobación del módulo 5.1.7

Ahora ya podrá entrar los nombres de variables y especificar una serie de valores para cada uno de ellos. Entre una fórmula compuesta por números, símbolos matemáticos y variables que haya definido. Detenga el programa y compruebe que esto se ha traducido correctamente. Deberá encontrarlo guardado en la variable alfanumérica temporal G$ y como un elemento en la tabla E$. Si conoce la respuesta correcta para su fórmula particular, puede comprobar la exactitud de los módulos entrando PRINT VAL G$ en modo directo. También podrá utilizar los resultados de una fórmula como variable en otra, utilizando variables LINEA.

MODULO 5.1.8

```
1400 REM ************************
1410 REM CALCULAR TABLA
1420 REM ************************
1430 INPUT "CUANTAS COLUMNAS? ";
COLUMNAS
1440 FOR K=1 TO LINEAS: LET F$=C
$(K): GO SUB 1490: LET E$(K)=G$:
 NEXT K
1450 FOR I=1 TO LINEAS: FOR J=1
TO COLUMNAS
1460 LET D(I,J)=VAL E$(I)
1470 NEXT J: NEXT I
1480 RETURN
```

Una vez evaluados los resultados de la fórmula definida por el usuario, utilizando los valores especificados para las variables entradas, quizá desee modificar una variable determinada. En lugar de reevaluarlo todo cada vez que se modifica una variable, este módulo recalcula la totalidad de la tabla de «líneas y columnas» cuando el usuario lo especifica. Las fórmulas no traducidas se consiguen de la tabla C$ y se vuelven a traducir basándose en los últimos datos.

Comentario

Línea 146∅. A primera vista quizá parezca que esta línea coloque una serie completa de elementos de la tabla D al mismo valor, pero esto no es cierto. La evaluación de E$(I) cambiará cada vez que cambie J, ya que hemos traducido las variables definidas por el usuario a la forma B(X,J) o D(X,J). Observe que no se hace ningún intento de resolver los problemas que se plantean por la interacción de los valores de LINEAS. Si el valor de LINEA1 depende del valor de LINEA 2 y LINEA2 debe modificarse por este bucle, el valor de LINEA 1 no tendrá en cuenta este hecho a menos que el módulo se llame dos veces. Las interacciones más complejas entre las líneas deberán pensarse antes de entrarlas.

MODULO 5.1.9

```
1300 REM ************************
1310 REM VISUALIZAR TABLA
1320 REM ************************
1330 INPUT "COLUMNA A VISUALIZAR
? (0=SALIR)";COLUMNA: PAPER 7: I
F COLUMNA=0 THEN PAPER 7: RETURN
1340 CLS : PRINT "LINEAS      VAL
ORES"
1350 FOR I=1 TO LINEAS: PAPER (5
+2*(I/2=INT (I/2))): PRINT A$(I)
```

```
;" ";
1360 LET M$=STR$ D(I,COLUMNA): I
F LEN M$>10 THEN PRINT M$( TO 10
): GO TO 1380
1370 PRINT M$
1380 NEXT I
1390 GO TO 1330
```

Este módulo visualiza el breve título de cada línea o fórmula y el valor que se deriva de la línea al utilizar una de las diez columnas posibles de valores de las variables.

Comentario

Líneas 136∅-137∅. Estas dos líneas truncan cualquier cifra a 1∅ dígitos. El único propósito de esto es permitir visualizar dos columnas con un mínimo de cambios en el programa. Si va a necesitar 1∅ o más dígitos de una forma regular, quizá tenga que modificar esto.

Comprobación del módulo 5.1.9

Ahora ya podrá comprobar el programa completo entrando 2∅ variables, cada una con 1∅ posibles valores, hasta 2∅ líneas de fórmulas que utilicen estas variables y breves títulos para cada una de estas líneas y visualizar los resultados, columna a columna. Si estas comprobaciones son satisfactorias el programa ya está listo para utilizarlo.

Resumen

Como intento de programa de propósito general, a primera vista éste parece poco interesante. Sin embargo, puede ser de mucha utilidad en una gran variedad de campos. En los negocios puede utilizarse para examinar los efectos de los cambios en los costes de los materiales o en los precios o en ambas cosas. En el hogar puede ser un adjunto a los programas financieros del capítulo 2, realizando los cálculos para los impuestos, si puede encontrar la fórmula adecuada. Los aficionados quizá deseen utilizarlo para cálculos repetitivos en otros campos. Con muchos otros programas de este libro, se trata de una herramienta flexible y tan sólo podrá obtener una opinión adecuada de sus posibilidades jugando con él, aplicándolo y modificándolo según sus propias necesidades.

Posibles mejoras

1) Si no necesita más de tres o cuatro dígitos para algunas aplicaciones, adapte el programa para que pueda visualizar varias columnas a la vez.
2) Si posee un Spectrum 48 K, amplíe el programa para que pueda cubrir más variables y fórmulas, proporcionando así la oportunidad de construir una biblioteca de fórmulas útiles.

5.2 Calorías

Tal como está este programa puede serle útil o no. Si necesita contar las calorías, puede ahorrarle mucho tiempo y hacer que sus esfuerzos den resultados mucho más precisos. Pero tanto si le interesan las calorías como si no, este programa merece un estudio cuidadoso ya que es un ejemplo de la forma en que el estilo modular de programación que hemos adoptado permite adaptar rápidamente el material escrito con anterioridad para otras utilizaciones. Este programa, en su mayoría, es una copia disfrazada de gran parte de las secciones del Test de Respuesta Múltiple. La brevedad de los comentarios en el programa le dará cierta idea de la velocidad en que fue escrito. A pesar de este hecho es un programa importante en el sentido de que muestra cómo pueden construirse y utilizarse diccionarios de elementos y de sus cantidades relacionadas.

MODULO 5.2.1

```
1180>REM ************************
1190 REM MENU
1200 REM ************************
1210 BORDER 4: INK 0: PAPER 7: C
LS : PRINT "          MENU"
1220 PRINT '"1)VISUALIZAR LA LIS
TA DE HOY"
1230 PRINT '"2)ENTRADA A LA LIST
A DE HOY"
1240 PRINT '"3)INICIAR LISTA NUE
VA"
1250 PRINT '"4)BORRAR DE LA LIST
A DIARIA"
1260 PRINT '"5)EXTENDER DICCIONA
RIO"
1270 PRINT '"6)VISUALIZAR DICCIO
NARIO/BORRAR"
1280 PRINT '"7)INICIALIZAR"
1290 PRINT '"8)FINALIZAR"
1300 INPUT "Cual requiere?";Z$:
CLS
1310 IF Z$="1" THEN GO SUB 1430
1320 IF Z$="2" THEN GO SUB 1580
1330 IF Z$="3" THEN GO SUB 1700
```

```
1340 IF Z$="4" THEN GO SUB 2520
1350 IF Z$="5" THEN GO SUB 1740
1360 IF Z$="6" THEN GO SUB 2170
1370 IF Z$="7" THEN GO TO 1030
1380 IF Z$="8" THEN GO TO 1400
1390 CLS : GO TO 1180
1400 PRINT FLASH 1;AT 10,12;"CAL
ORIAS"
1410 INPUT "HA ENTRADO INFORMACI
ON NUEVA QUE DESEE GUARDAR? (S/N
)";Q$: IF Q$="S" THEN SAVE "CALO
RIAS": BEEP 1,40: PRINT '"REBOBI
NAR, UNA TECLA PARA VERIFICAR":
PAUSE 0: VERIFY "CALORIAS": PRIN
T '"VERIFICADO"
1420 STOP
```

Un módulo estándar de menú

MODULO 5.2.2

```
   1>GO TO 1180
1000 REM ***********************
1010 REM VARIABLES
1020 REM ***********************
1030 DIM A$(500,15)
1040 DIM B$(500,15)
1050 DIM C(500)
1060 DIM T$(50,2,20)
1070 DIM O$(32)
1080 DIM T(50)
1090 LET LISTA=0
1100 LET ELEMFIN=3
1110 LET ELEMENTOS=2
1120 LET I$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 2
1130 LET A$(2,1)=CHR$ 255
1140 LET P$="          "
1150 DIM F$(15)
1160 DIM G$(17)
1170 LET L$="*******************
*************"
```

¿Por qué he puesto las variables antes del menú? Bueno, me pareció una buena idea cuando lo hice. La utilización de las distintas variables se analizará en el transcurso del programa.

MODULO 5.2.3

```
1740>REM ***********************
1750 REM ENTRAR ELEMENTOS
1760 REM ***********************
1770 CLS : PRINT "NUEVOS ELEM. P
ARA EL DICCIONARIO"
1780 PRINT '"COMIDA:": INPUT "NO
MBRE O ""ZZZ"" PARA SALIR ";F$:
IF F$( TO 3)="ZZZ" THEN RETURN
1790 PRINT AT 2,8;F$
1800 PRINT '"UNIDADES:": INPUT "
```

```
NOMBRE ";G$
1810 PRINT AT 4,10;G$
1820 PRINT '"CALORIAS POR UNIDAD
:": INPUT "NUMERO ";H$
1830 LET TEMP=VAL H$
1840 PRINT AT 6,21;H$
1850 INPUT "Son correctas? (S/N)
";Q$: IF Q$="N" THEN GO TO 1770
1860 CLS : GO SUB 1880: GO SUB 2
030
1870 LET ELEMENTOS=ELEMENTOS+1:
GO TO 1740
```

Este módulo tan directo acepta tres entradas que se refieren al nombre de un alimento, las unidades en que se mide habitualmente y el número de calorías por unidad.

MODULO 5.2.4

```
1880>REM ***********************
1890 REM BUSQUEDA BINARIA
1900 REM ***********************
1910 DEF FN A()=256*CODE I$(2*S-
1)+CODE I$(2*S)
1920 LET POTEN=INT (LN (ELEMENTO
S)/LN 2)
1930 LET S=2↑POTEN
1940 FOR I=POTEN-1 TO 0 STEP -1
1950 LET P=FN A()
1960 LET S=S-(2↑I)*(A$(P)>F$)+(2
↑I)*(A$(P)<F$)
1970 IF S<2 THEN LET S=2
1980 IF S>ELEMENTOS-1 THEN LET S
=ELEMENTOS-1
1990 NEXT I
2000 LET P=FN A()
2010 IF A$(P)>F$ THEN LET S=S-1
2020 RETURN
```

Este es un módulo de búsqueda binaria para determinar la situación de un alimento dentro del diccionario total de alimentos, guardado en A$. Pueden guardarse hasta 5ØØ alimentos, cada uno con un nombre de 15 caracteres de longitud.

MODULO 5.2.5

```
2030>REM ***********************
2040 REM INSERCION
2050 REM ***********************
2060 IF LEN P$=4 THEN GO TO 2100
2070 LET LUGAR=256*CODE P$(3)+CO
DE P$(4)
2080 LET P$=P$( TO 2)+P$(5 TO )
2090 GO TO 2120
2100 LET LUGAR=ELEMFIN
2110 LET ELEMFIN=ELEMFIN+1
2120 LET A$(LUGAR)=F$
```

```
2130 LET B$(LUGAR)=G$
2140 LET C(LUGAR)=TEMP
2150 LET I$=I$( TO 2*S)+CHR$ INT
 (LUGAR/256)+CHR$ INT (LUGAR-256
*INT (LUGAR/256))+I$(2*S+1 TO )
2160 RETURN
```

Este módulo inserta el nuevo elemento en su lugar correcto, siguiendo el orden utilizado en el módulo equivalente del programa de Respuesta Múltiple. Primeramente se rellenan los espacios vacíos de la lista, y a continuación las posiciones situadas al final de la lista. El orden real viene indicado por un par de caracteres contenidos en la tabla de punteros, I$.

Comprobación del módulo 5.2.5

Por el momento, aunque el diccionario no pueda visualizarse fácilmente, deberá ser capaz de entrar elementos en este diccionario. Estos se guardarán en A$ en el orden en que han sido entrados. Las unidades asociadas y los valores caloríficos deberán guardarse en las posiciones paralelas de B$ y C, respectivamente. El orden correcto de los elementos deberá poderse descifrar a partir del valor del código representado por el par de caracteres en I$.

MODULO 5.2.6

```
2170>REM ************************
2180 REM BUSQUEDA DE USUARIO
2190 REM ************************
2200 PRINT "                 VISUAL
IZAR"
2210 PRINT '"AL APARECER ELEMENT
O, ENTRAR:"
2220 PRINT '">NUMERO POSITIVO O
NEGATIVO PARA MOVER PUNTERO"
2230 PRINT '">""DDD"" PARA BORRA
R ELEMENTO"
2240 PRINT '">""ZZZ"" PARA ABAND
ONAR FUNCION"
2250 PRINT L$
2260 IF ELEMENTOS=2 THEN RETURN
2270 LET S=2
2280 LET P=FN A()
2290 PRINT AT 12,0;"REGISTRO No.
 ";S-1
2300 PRINT '"COMIDA:";A$(P)
2310 PRINT '"UNIDADES:";B$(P)
2320 PRINT '"CALORIAS:";C(P);"
"
2330 INPUT "Cual requiere?";S$
2340 IF S$="DDD" THEN GO SUB 244
0: RETURN
2350 IF S$="ZZZ" THEN RETURN
2360 IF S$<>"" THEN GO TO 2400
2370 LET S=S+1
```

```
2380 IF S=ELEMENTOS THEN RETURN
2390 GO TO 2280
2400 LET S=S+VAL S$
2410 IF S>=ELEMENTOS THEN RETURN
2420 IF S<2 THEN LET S=2
2430 GO TO 2280
```

Este es un sencillo módulo de búsqueda basado en el que utilizó en el programa de Respuesta Múltiple. No es realmente una búsqueda; el usuario tan sólo puede recorrer, hacia adelante o hacia atrás, el archivo.

El módulo de borrado se llama también desde este módulo.

Comprobación del módulo 5.2.6

Ahora ya podrá visualizar todos los alimentos entrados mediante una orden.

MODULO 5.2.7

```
2440>REM ************************
2450 REM ELIMINACIONES
2460 REM ************************
2470 LET P=FN A(): LET A$(P)="":
 LET B$(P)="": LET C(P)=0
2480 LET I$=I$( TO 2*S-2)+I$(2*S
+1 TO )
2490 LET ELEMENTOS=ELEMENTOS-1
2500 LET P$=P$( TO 2)+CHR$ INT (
P/256)+CHR$ INT (P-256*INT (P/25
6))+P$(3 TO )
2510 RETURN
```

Este es el método de borrado utilizado en el programa de Respuesta Múltiple. La dirección del elemento borrado se elimina de I$ y el espacio que deja se guarda en P$. Observe que el registro de hecho no se ha eliminado realmente de la tabla. Lo que sucede es que el programa ya no lo tendrá en cuenta.

Comprobación del módulo 5.2.7

Ahora ya podrá borrar elementos del diccionario.

MODULO 5.2.8

```
1580>REM ************************
1590 REM ANADIR A LA LISTA DE
     HOY
```

```
1600 REM ***********************
1610 PRINT "    ANADIR A LA LISTA
 DE HOY"
1620 PRINT '"COMIDA:": INPUT F$:
 PRINT AT 2,8;F$
1630 GO SUB 1880: IF A$(P)<>F$ T
HEN PRINT AT 10,0;"*COMIDA ESPEC
IFICADA DESCONOCIDA, COMPROBAR P
OR FAVOR.": PAUSE 200: RETURN
1640 PRINT '"UNIDADES:";B$(P)
1650 PRINT '"CANTIDAD:": INPUT Q
: PRINT AT 6,10;Q
1660 INPUT "Son correctas? (S/N)
";Q$: CLS : IF Q$="N" THEN GO TO
 1580
1670 LET LISTA=LISTA+1: LET T$(L
ISTA,1)=A$(P): LET T$(LISTA,2)=S
TR$ Q+" "+B$(P): LET T(LISTA)=Q*
C(P)
1680 INPUT "Mas elementos? (S/N)
";Q$: CLS : IF Q$="S" THEN GO TO
 1580
1690 RETURN
```

Con este módulo nos alejamos de la estructura que hemos tomado del programa de Respuesta Múltiple y llegamos a las partes nuevas de nuestra nueva aplicación. El propósito de este módulo es el crear de una forma rápida una lista diaria de alimentos, basada en los alimentos que ya están guardados en el diccionario. La lista no tiene por qué realizarse de una vez sino a medida que se van comiendo los alimentos o se planean los menús.

Línea 163∅. Observe la interesante utilización que se hace aquí del módulo de búsqueda binaria. Cuando se entra un alimento para lista diaria, el nombre se envía al módulo de búsqueda que identifica la posición que este alimento ocuparía si estuviese contenido en el diccionario. Al volver de esta búsqueda, el alimento que está en esta posición se compara con el alimento entrado para la lista diaria. Si no son iguales quiere decir que el alimento entrado para la lista diaria no está contenido en el diccionario, de lo cual se informa al usuario.

Línea 167∅. La lista diaria actual se guarda en las dos mitades de la tabla T$ y los valores caloríficos asociados en T.

Comprobación del módulo 5.2.8

Ahora el programa deberá aceptar la entrada de alimentos en la lista diaria, visualizar las unidades usuales de medida e invitarle a entrar una cantidad. Los alimentos que no estén en el diccionario no serán aceptados.

MODULO 5.2.9

```
1430 REM ***********************
1440 REM LISTA DE HOY
1450 REM ***********************
1460 LET TOTAL=0
1470 FOR I=1 TO LISTA
1480 PRINT "COMIDA   :";T$(I,1)
1490 PRINT "CANTIDAD:";T$(I,2)
1500 PRINT "CALORIAS:";T(I)
1510 PRINT L$
1520 LET TOTAL=TOTAL+T(I)
1530 NEXT I
1540 PRINT '"TOTAL CALORIAS:";TO
TAL
1550 PRINT '"Pulsar cualquier te
cla para con."
1560 PAUSE 0
1570 RETURN
```

Este módulo visualiza simplemente la lista de alimentos del día, dando su nombre, la cantidad y el total de calorías para esta cantidad. Al final de la lista se visualizan el total de calorías para esta lista.

MODULO 5.2.10

```
1700 REM ***********************
1710 REM NUEVA LISTA
1720 REM ***********************
1730 DIM T$(50,2,20): DIM T(50):
 LET LIST=0: GO SUB 1580: RETURN
```

Este módulo permite inicializar las tablas relativas a la lista diaria sin afectar a las tablas principales.

MODULO 5.2.11

```
2520 REM ***********************
2530 REM BORRAR DE LA LISTA
2540 REM ***********************
2550 FOR I=1 TO LISTA
2560 PRINT T$(I,1)
2570 INPUT "DDD=BORRAR   /ENTER=
SIGUIENTE    /ZZZ=SALIR";Q$: IF Q
$="ZZZ" THEN RETURN
2580 IF Q$="DDD" THEN GO TO 2600
2590 NEXT I: RETURN
2600 LET T$(I,1)="": LET T$(I,2)
="": LET T(I)=0
2610 FOR J=1 TO LISTA-1
2620 LET T$(J,1)=T$(J+1,1)
2630 LET T$(J,2)=T$(J+1,2)
2640 LET T(J)=T(J+1)
2650 NEXT J
2660 LET LISTA=LISTA-1
2670 RETURN
```

Este módulo borra elementos de la lista diaria.

Comprobación del módulo 5.2.11

Ahora podrá establecer una lista diaria, visualizarla y borrar elementos de la misma. También podrá inicializar las tablas para empezar una nueva lista. Si estas comprobaciones son satisfactorias el programa ya está preparado para su utilización.

Resumen

Espero que durante la ejecución de este programa se haya dado cuenta de la ventaja que representa el planteo modular en la escritura de programas. Ha permitido que la mayor parte de este programa se sacase simplemente de una aplicación muy distinta. A medida que pase el tiempo y cuando haya desarrollado sus propios métodos favoritos para tratar los distintos tipos de materiales y procesos llegará a tener cierta habilidad para encontrar métodos de realizar una nueva aplicación sobre el Spectrum utilizando otros métodos que ya ha conocido con anterioridad y líneas de programa que ha comprobado y depurado anteriormente.

De esto debe sacar la importante lección de que es más importante el dominar métodos que programas, aunque los programas a veces suelen contener nuevos métodos. Una buena biblioteca de programas le dejarán en una buena posición hasta que aparezca una aplicación totalmente nueva. Una buena colección de métodos, contenidos en módulos claramente identificables, no le dejarán nunca colgado. Por lo tanto, no se limite a métodos que necesite tan sólo para las aplicaciones actuales. Si ve una forma interesante de hacer las cosas en una revista o un libro, escriba un sencillo programa para utilizarla, aunque sea sólo por hacerlo. Estará ya preparada para cuando tenga que utilizarla.

Entre tantas generalizaciones no pierda de vista el hecho de que este programa tiene un amplio campo de aplicaciones potenciales. Los alimentos no son en absoluto la única área donde es útil el tener un diccionario de elementos, asociados con sus unidades típicas y un valor de algún tipo. Los tenderos, por ejemplo, generalmente tienen en su stock menos de 5ØØ productos, cada uno se vende en una unidad de algún tipo y a cierto precio. ¿Por qué no adaptar el programa para una utilización de este tipo? Recuerde que lo que cuenta son los métodos y no los programas. Este programa es un ejemplo de un método, usted ya me entiende...

Posibles mejoras

1) Una diferencia importante entre este programa y el de Respuesta Múltiple es que este último dimensionaba sus propias tablas para poder tratar con varias aplicaciones mientras utilizaba la memoria al máximo. Podría cambiar el programa de forma que se convirtiera en una herramienta de propósito general para nombre y cantidad, con los nombres de los elementos especificados por el usuario.

5.3 Gráficas

Este pequeño programa es una herramienta de dibujo de gráficas de propósito general que permite al usuario el definir las unidades de ambos ejes con respecto al nombre y su longitud y entrar los datos de forma ordenada o desordenada para crear una gráfica lineal.

MODULO 5.3.1

```
1000>REM ************************
1010 REM MENU
1020 REM ************************
1030 CLS : PRINT AT 0,12; FLASH
1; INK 2;"GRAFICAS"
1040 PRINT '"1)DISPONER PRESENTA
CION"
1050 PRINT '"2)ENTRAR VALORES"
1060 PRINT '"3)DIBUJAR GRAFICA"
1070 PRINT '"4)INICIALIZAR"
1080 PRINT '"5)FINALIZAR"
1090 INPUT Z$: CLS
1100 IF Z$="1" THEN GO SUB 1180
1110 IF Z$="2" THEN GO SUB 1350
1120 IF Z$="3" THEN GO SUB 1530
1130 IF Z$="4" THEN CLEAR : LET
Z$="1"
1140 IF Z$="5" THEN GO TO 1160
1150 GO TO 1000
1160 PRINT AT 10,12; FLASH 1; IN
K 2;"GRAFICAS": INPUT "DESEA GRA
BAR?";Q$: IF Q$="S" THEN SAVE "G
RAFICA": INPUT "REBOBINAR, ""ENT
ER"" PARA         VERIFICAR.";Q$
: LET Q$="S": VERIFY "GRAFICA":
PRINT '"       PROGRAMA VERIFICADO
."
1170 STOP
```

Otro módulo estándar de menú. Observe la línea 113∅: Z$ tiene que ponerse a “1” después de que la memoria haya sido borrada (CLEAR) o se generará un mensaje de error en la línea 113∅ ya que no existirá Z$.

```
1180>REM ************************
1190 REM DISPONER EJES
1200 REM ************************
1210 PRINT "CUANTOS INTERVALOS E
N EL EJE    HORIZONTAL? ";: INPU
T H: PRINT H: LET LH=INT (236/H)
1220 PRINT "CUANTOS INTERVALOS E
N EL EJE     VERTICAL? ";: INPUT
V: PRINT V: LET LV=INT (156/V)
1230 CLS : PRINT AT 8,3; FLASH 1
; INK 2;"LOS EJES APARECERAN ASI
"
1240 LET VMARK=300: LET HMARK=30
0: GO SUB 1450
1250 INPUT "PUEDE PONER UNA MARC
A CADA      CIERTO NUMERO DE ESP
ACIOS.      ENTRE ESPACIADO HORI
ZONTAL:";HMARK
1260 INPUT "ESPACIADO VERTICAL:"
;VMARK
1270 CLS : GO SUB 1450
1280 INPUT "ES SATISFACTORIO? (S
/N)";Q$: IF Q$="N" THEN CLS : GO
 TO 1180
1290 INPUT "NOMBRE DE LAS UNIDAD
ES HORIZ.? ";H$
1300 INPUT "NOMBRE DE LAS UNIDAD
ES VERT.? ";V$
1310 INPUT "VALOR MINIMO DEL EJE
 VERT.? ";BASE
1320 PRINT '"MAXIMO VALOR REPR.
POR ";V;" UNID.": INPUT "OTRO VA
LOR? ";LIMIT
1330 LET VVAL=(LIMIT-BASE)/V
1340 DIM G(H): FOR I=1 TO H: LET
 G(I)=-999: NEXT I: RETURN
```

El propósito de este módulo es permitir al usuario que pueda especificar cuántos intervalos habrá en los ejes horizontal y vertical y dar nombre a las unidades involucradas.

Comentario

Línea 121∅. El eje horizontal tendrá 236 pixels de largo. La variable LH (longitud horizontal) se coloca a 236 dividido por el número deseado de intervalos. Generalmente esto dará como resultado el que haya algunos intervalos más de los que haya especificado el usuario.

Línea 122∅. Se crea una variable similar para el eje vertical.

Líneas 124∅-128∅. Las unidades quedarán marcadas sobre los ejes mediante pequeñas líneas realizadas con la orden DRAW. En aras de una mayor claridad algunas de las líneas pueden ser ligeramente más largas, por ejemplo cada 5 unidades puede aparecer una más resaltada. Los intervalos en los que sucederá esto se guardan en HMARK y VMARK. Estas variables se inicializan a 3∅∅ cuando se dibujan los ejes por primera vez, de forma que no haya ninguna marca

resaltada en ninguno de los dos ejes. Una vez visualizados los ejes básicos se invita al usuario a que especifique los intervalos para estas marcas más resaltadas y después los ejes vuelven a visualizarse para pedir confirmación.

Líneas 129∅-132∅. Se piden los nombres de las unidades de ambos ejes. Se supone que las unidades del eje horizontal empiezan desde cero, pero para el eje vertical se pide al usuario que especifique el valor que corresponde al inicio y el valor que corresponde al final. A partir de esto se calcula el valor de una unidad individual sobre el eje vertical, y se guarda en la variable VVAL. Las unidades del eje horizontal se supone que tienen el valor 1.

Línea 134∅. La tabla G se dimensiona dejándola preparada para los posibles elementos H de los datos.

Comprobación del módulo 5.3.2

Los ejes no pueden dibujarse hasta que se haya entrado el siguiente módulo, pero insertando una línea temporalmente, 145∅ RETURN, el módulo deberá pedir el número de intervalos, las señales que deben remarcarse y el nombre de las unidades.

MODULO 5.3.3

```
1450>REM ************************
1460 REM DIBUJAR EJES
1470 REM ************************
1480 FOR I=1 TO 19: PRINT AT I,1
;"▌": NEXT I: PRINT AT 20,1;"▙";
1490 FOR I=1 TO 29: PRINT "▄";:
NEXT I
1500 LET C=0: FOR I=11+LH TO 247
 STEP LH: LET C=C+1: PLOT INVERS
E 1; OVER 1;I,13: DRAW OVER 1;0,
-5-3*(C/HMARK=INT (C/HMARK)): NE
XT I
1510 LET C=0: FOR I=11+LV TO 167
 STEP LV: LET C=C+1: PLOT INVERS
E 1; OVER 1;13,I: DRAW OVER 1;-5
-3*(C/VMARK=INT (C/VMARK)),0: NE
XT I
1520 RETURN
```

Este módulo dibuja los ejes de la gráfica.

Comentario

Línea 15∅∅. El eje horizontal se rellena con líneas perpendiculares al mismo, de 5 pixels de largo y separadas LH pixels. A cada

intervalo en que deba resaltarse la marca, la línea tiene tres pixels más.

Línea 151∅. Un proceso similar se repite para el eje vertical.

Comprobación del módulo 5.3.3

Ahora ya debe poder entrar los detalles de las unidades y los nombres de las unidades y obtener la visualización de los ejes para su confirmación.

MODULO 5.3.4

```
1350>REM ************************
1360 REM ENTRAR VALORES
1370 REM ************************
1380 PRINT "VALOR DE ";H$;"? ";:
 INPUT H1: PRINT H1
1390 PRINT '"VALOR DE ";V$;"? ";
: INPUT V1: PRINT V1
1400 PRINT '"SON CORRECTOS?": IN
PUT Q$: IF Q$="N" THEN CLS : GO
TO 1380
1410 IF G(H1)<>-999 THEN PRINT '
"ESTA POSICION YA ESTA OCUPADA
 POR EL VALOR ";G(H1);"."'"DESEA
 REEMPLAZARLO? (S/N)": INPUT Q$:
 IF Q$="N" THEN CLS : GO TO 1350
1420 LET G(H1)=V1: INPUT "OTRO V
ALOR?";Q$: IF Q$="S" THEN CLS :
GO TO 1350
1430 RETURN
```

Este módulo acepta los datos que se utilizarán para dibujar la propia gráfica.

Comentario

Línea 141∅. Al inicializar la tabla G se ha rellenado con el valor -999 para indicar elementos vacíos, ya que el cero no es apropiado porque puede utilizarse frecuentemente. Si se entra un elemento para una posición que no contenga -999, se informa al usuario de que ya se ha entrado un dato para esta posición.

Línea 142∅. El dato se coloca en la tabla G. No se hace ninguna comprobación para ver si cae dentro de los límites especificados para la gráfica, aunque se invita al usuario a que confirme el dato en la línea 14∅∅. Obsérvese que ya que las unidades sobre el eje horizontal se supone que aumentan de uno en uno a partir de 1, la posición de un elemento en el eje horizontal es también su posición en G.

Comprobación del módulo 5.3.4

La gráfica todavía no puede dibujarse, pero el módulo debe aceptar datos e informarle si está reescribiendo un dato que ya entró anteriormente.

MODULO 5.3.5

```
1530>REM *********************
1540 REM DIBUJAR GRAFICA
1550 REM *********************
1560 PRINT AT 0,0;V$;" BASE:";BA
SE;"/LIMITE:";LIMIT: FOR I=1 TO
LEN H$: PRINT AT 21-LEN H$+I,31;
H$(I): NEXT I
1570 GO SUB 1450
1580 FOR I=1 TO H: IF G(I)=-999
THEN NEXT I: PRINT AT 10,3;"NO H
AY DATOS.": STOP
1590 PLOT INVERSE 1; OVER 1;11+I
*LH,11+(G(I)-BASE)/VVAL*LV
1600 FOR J=I TO H: IF G(J)=-999
THEN GO TO 1640
1610 FOR K=J+1 TO H
1620 IF G(K)=-999 THEN NEXT K: I
F K>H THEN GO TO 1650
1630 DRAW LH*(K-J),(G(K)-G(J))/V
VAL*LV
1640 NEXT J
1650 INPUT """ENTER"" PARA CONTI
NUAR";Q$: RETURN
```

Este módulo dibuja la gráfica basándose en los datos suministrados por el usuario.

Comentario

Línea 156∅. Los nombres de las unidades para los ejes horizontal y vertical se visualizan en los lugares adecuados.

Línea 158∅. Si todavía no hay presente ningún dato, el programa informa de ello al usuario.

Línea 159∅. Esta línea quizá sea un poco confusa a primera vista. Lo que hace es trazar un punto invisible en el punto donde la gráfica debe empezar. 11+I*LH es la posición horizontal correspondiente al primer elemento de los datos en la tabla G. G(I)-BASE es el valor de un elemento de los datos en relación con el punto inicial del eje vertical. (G(I)-BASE)/VVAL es el número de unidades que éste debe representar sobre el eje vertical. (G(I)-BASE)/VVAL*LV es la altura en pixels del punto determinado.

Líneas 16∅∅-164∅. Se necesitan dos bucles, uno para registrar la posición del último punto trazado y el otro para encontrar la siguiente posición que debe dibujarse. No hay ninguna necesidad de que el

usuario rellene todos los espacios destinados a los datos, por lo que el módulo saltará los elementos vacíos de G. Una vez se ha encontrado un dato, la línea 163Ø dibuja una línea hasta el punto que corresponde horizontalmente a la posición del dato de G, teniendo en cuenta cualquier espacio vacío que haya entre medio, y verticalmente hasta el número de pixels por encima o por debajo del valor previamente trazado. (G(K)-G (J))/VVAL*LV es directamente equivalente a la expresión explicada en la línea 159Ø.

Comprobación del módulo 5.3.5

Si ha entrado algunos datos adecuados, ahora ya podrá dibujar una gráfica. Si esto se realiza satisfactoriamente el programa ya está listo para su utilización.

Resumen

Es poco probable que la mayoría de los usuarios del Spectrum quieran utilizar un programa como éste cada día. Pero es un ejemplo del tipo de herramienta que puede tener cierto valor al añadirlo a una biblioteca de programas, sabiendo que un día quizá quiera dibujar una gráfica, pero no hasta el punto de que deba escribir un programa especial para ello. Quizá también haya observado, al entrar el programa, que aunque es un programa bastante rudimentario, el añadirle pequeños toques, como el resaltar algunas marcas en los ejes, la visualización de los nombres de las unidades y el punto inicial del eje vertical, hacen diferenciar claramente un gráfico que es meramente correcto, de uno que es comprensible y correcto. Después de todo, ¿para qué nos sirve guardar algo si la próxima vez que lo analicemos no podemos recordar qué es lo que quería representar o en qué unidades está representado?

Posibles mejoras

1) Un proyecto relativamente sencillo es adaptar el programa para que pueda dibujar también gráficos de barras. Las barras podrían ser de grueso variable, de una aplicación a otra, por lo que el único cambio que debería realizarse en el programa sería en el módulo que dibuja la gráfica. Una vez establecido el valor para una posición determinada, el módulo tendría que dibujar LH líneas hasta la base horizontal, o quizá LH-1 para que quede marcada una separación entre las barras.

2) También sería de utilidad si el programa pudiera imprimir el gráfico sin tener que detener el programa para entrar la orden COPY en modo directo.

5.4 Renumeración

Aunque he sugerido que la mayoría de los propietarios del Spectrum utilizarán muy poco el último programa, hay que decir lo contrario para este programa, ya que probablemente se utilizará más que cualquier otro de nuestra biblioteca de programas. Para aquellos que no están familiarizados con la idea, un programa de renumeración es uno que cambia los números de línea irregulares que van apareciendo cuando se escriben la mayoría de los programas, por una estructura regular como la que se ve en la mayoría de los programas de este libro.

Esto no es una tarea tan difícil como pueda parecer. Los números de línea están guardados al principio de cada línea en dos bytes — de la misma forma que hemos guardado valores mediante dos caracteres en cadena. Todo lo que hay que hacer es encontrar la posición del número de línea en la que empieza el programa, colocar mediante POKE en las dos posiciones el número de línea con el que se quiere empezar, después seguir a través de los números de línea, colocando mediante POKE números que aumenten con intervalos regulares. Puede hacerse esto en tres o cuatro líneas.

El problema es que la renumeración no se acaba aquí debido a que hay que tener en cuenta los GOTO y los GOSUB. Repartidos a lo largo del programa aparecerán referencias a los números de línea originales. Si se cambian estos números de línea, entonces se produciría el caos. Por lo tanto tendremos que renumerar los GOTO y los GOSUB para que coincidan con los números de línea cambiados. Esto no es tan fácil. La forma en que los destinos del GOTO y del GOSUB se guardan es considerablemente distinta y más compleja que la forma en que se guardan los números de línea. Ya hemos indicado anteriormente que los valores numéricos generalmente requieren cinco bytes de la memoria del Spectrum para abarcar el amplio rango de números que el Spectrum puede expresar con exactitud. Desafortunadamente, en algunos aspectos los destinos del GOTO y del GOSUB utilizan esta forma más compleja de almacenaje, aunque los 9999 números de línea posibles realmente tan sólo requieren dos caracteres para ser guardados.

En algunos aspectos esto aumenta la flexibilidad de la máquina —ya que el destino se considera como un número real, puede expresarse en cualquiera de las formas en que el Spectrum reconocerá un

número. Por ejemplo, 1ØØØ+X,2*A/Y, VAL"25ØØ" serían todos destinos válidos con tal de que sus resultados estuvieran comprendidos entre 1 y 9999. Hay algunos trucos interesantes en el BASIC de Sinclair que dependen de esta capacidad. Sin embargo, a la hora de la renumeración, la representación con cinco bytes es un problema. El problema todavía empeora por el hecho de que el número del destino, al igual que cualquier número real visualizado en una línea de programa, está de hecho registrado dos veces en la línea, una vez como un conjunto de caracteres para que usted los vea y una segunda vez en la forma de cinco bytes.

La conclusión de todo esto es que no sólo tenemos que renumerar la línea, sino que también tenemos que saber si hay algún GOTO o GOSUB que esté señalando a la línea que se está renumerando. Si es así tendremos que cambiar dos números más mediante métodos distintos.

Dentro de ciertos límites esto es exactamente lo que hace este pequeño programa, y aunque de ningún modo será tan rápido y conveniente como un buen programa en código máquina para conseguir el mismo resultado, mezclado con sus propios programas hará un trabajo adecuado hasta que se decida a comprar un programa comercial mucho más caro o escriba uno usted mismo.

```
9958 STOP
9959 LET T$="": LET X=23635
9960 DEF FN A(X)=PEEK X+256*PEEK
 (X+1)
9961 DEF FN B(S)=256*PEEK S+PEEK
 (S+1)
9962 LET S=FN A(X)
9963 LET LINEA=FN B(S): IF LINEA
>=9958 THEN GO TO 9976
9964 LET LONG=FN A(S+2)
9965 IF PEEK (S+4)=234 THEN GO T
O 9969
9966 FOR I=S+4 TO S+LONG+2
9967 IF PEEK I=236 OR PEEK I=237
 THEN GO SUB 9971
9968 NEXT I
9969 LET S=S+LONG+4
9970 GO TO 9963
9971 IF PEEK (I+5)=14 THEN GO TO
 9974
9972 PRINT "COMANDO NO ESTANDAR,
 LINEA ";LINEA
9973 STOP
9974 LET T$=T$+STR$ I+CHR$ PEEK
(I+1)+CHR$ PEEK (I+2)+CHR$ PEEK
(I+3)+CHR$ PEEK (I+4)
9975 RETURN
9976 LET BASE=1000: LET X=23635
9977 LET S=FN A(X)
9978 LET LINEA=FN B(S): IF LINEA
>=9958 THEN STOP
9979 LET LONG=FN A(S+2)
9980 FOR I=1 TO LEN T$ STEP 9
9981 IF VAL T$(I+5 TO I+8)=LINEA
 THEN GO SUB 9990
```

```
9982 NEXT I
9983 POKE S,INT (BASE/256)
9984 POKE S+1,BASE-256*INT (BASE
/256)
9985 LET BASE=BASE+10
9986 LET S=S+LONG+4
9987 GO TO 9978
9990 FOR J=1 TO 4
9991 POKE (VAL T$(I TO I+4)+J),C
ODE (STR$ BASE)(J)
9992 NEXT J
9993 LET BYTE1=128+INT (LN BASE/
LN 2+1)
9994 LET BYTE2=BASE*65536/(2↑(BY
TE1-128))
9995 LET MEMOR=VAL T$(I TO I+4)
9996 POKE MEMOR+6,BYTE1
9997 POKE MEMOR+7,INT (BYTE2/256
)-128
9998 POKE MEMOR+8,BYTE2-256*INT
(BYTE2/256)
9999 RETURN
```

El programa no está escrito en módulos ya que el objetivo es colocar el máximo posible en el menor número de líneas posible. Una vez haya comprendido el método, quizá desee acortar el programa utilizando líneas multisentencia.

Comentario

Línea 9958. Esta línea nos asegura que la ejecución normal de la parte principal del programa que se está renumerando no llegue nunca a la rutina de renumeración.

Línea 9959. T$ será utilizado para registrar cada aparición de un GOTO o un GOSUB en el programa, junto con su localización dentro de la memoria del Spectrum. X es la dirección en la memoria del Spectrum de la variable del sistema «PROG», que consiste en dos bytes que contienen la dirección inicial en memoria del programa actual en BASIC.

Línea 996Ø. Esta función debe considerarse como la dirección inicial del programa en BASIC actual. Obsérvese que cuando el Spectrum trabaja para sí, guarda los números de dos bytes de atrás hacia adelante — viene primero el menos significativo de los dos bytes.

Línea 9961. Para acabarle de confundir, los números de línea, que también tienen dos bytes, están guardados en el orden opuesto. Esta función convertirá el número de línea de dos bytes en una forma de representación más reconocible.

Línea 9962. En S se coloca la dirección del inicio del programa.

Línea 9963. En LINEA se coloca el número de línea actual, encontrado en este punto. Cuando la rutina ha pasado a través del programa hasta el punto donde se encuentra la rutina de renumeración, entonces esta sección del programa no se ejecuta.

Línea 9964. Los dos bytes que van a continuación del número de línea en cualquier línea contienen la longitud de la línea. Este valor puede utilizarse para pasar fácilmente hasta el principio de la línea siguiente.

Línea 9965. Las líneas que empiezan por REM (CHR$234) no son examinadas para buscar los GOTO y los GOSUB. Esto significa que si se tiene un GOTO calculado de la forma 1ØØØ*X, que el programa no podría tratar, tan sólo hay que colocar temporalmente un REM al principio de la línea y no causará problemas.

Líneas 9966-9968. La línea se examina byte a byte buscando los caracteres que representan el GOTO y el GOSUB (códigos 236 y 237 respectivamente).

Líneas 9969-997Ø. Si no se encuentra nada en la línea, la rutina pasa al principio de la línea siguiente.

Líneas 9971-9973. Si la ejecución ha llegado hasta este punto es debido a que se ha encontrado un carácter con el código 236 o 237 en esta línea. Estas líneas comprueban que se trata de un GOTO o un GOSUB normal, que la rutina puede tratar. Esto se hace teniendo en cuenta que cualquier destino estará representado visiblemente mediante cuatro dígitos, es decir, GOTO 1ØØØ, GOTO ØØØ1. Si esto es cierto, entonces el quinto carácter después del GOTO será un 14, que indica al Spectrum que a continuación hay un número de cinco bytes (la segunda de las dos representaciones del destino). Si éste no es el caso, la rutina se detiene. En muy pocas ocasiones la rutina se encontrará con bytes que contienen 236 o 237 y que no tienen nada que ver con un GOTO o un GOSUB y se detendrá. Una vez identificado el problema, la única solución es colocar temporalmente un REM al principio de la línea causante de los problemas.

Líneas 9974-9975. Si la rutina descubre un GOTO o un GOSUB estándar, guarda la dirección en T$, junto con los cuatro bytes que constituyen el destino.

Línea 9976. Cuando la rutina alcanza este punto, ya habrá ejecutado el programa una vez, guardando la posición de cada GOTO y cada GOSUB y las líneas a las que señalan. Ahora el programa inicializa una variable, BASE, que se utilizará como número de la primera línea.

Líneas 9977-9978. La rutina empieza de nuevo por el principio del programa y empieza a convertir los números de línea. Cuando se llegue a la línea 9958, la renumeración estará completa.

Líneas 998Ø-9982. Una vez establecido el número de línea, la rutina busca en T$ si se trata de una línea a la que apunta en algún lugar un GOTO o un GOSUB. Si así es, procede a renumerar el GOTO o el GOSUB en la forma explicada a continuación.

Líneas 9983-9987. El valor de BASE se coloca en la posición ocu-

pada por el antiguo número de línea. Se incrementa en 1∅ unidades la variable BASE y la rutina pasa al siguiente número de línea.

Líneas 999∅-9992. Si hay que renumerar un GOTO o un GOSUB, la rutina coloca primeramente los caracteres que constituyen la base en los cuatro bytes que hay a continuación del GOTO o del GOSUB.

Líneas 9993-9998. En esta sección pasamos a manipular la representación compleja mediante cinco bytes de los números que utiliza el Spectrum. Esta tarea se simplifica por el hecho de que tratamos únicamente con números positivos comprendidos entre 1 y 9999. Esto significa que tan sólo tenemos que tratar con los tres primeros bytes de los cinco. Hay dos tareas básicas. La primera es determinar el exponente del destino, una vez renumerado, cuando está expresado en binario —es decir, el número de lugares antes del punto decimal cuando el número está escrito en binario. El primer byte de cualquier número positivo guardado en el Spectrum será 128 más este exponente, y este valor se crea en la línea 9993.

Los dos bytes siguientes contienen la representación binaria del destino. Como ya tenemos el exponente que nos indica cuántos caracteres hay en este número binario, podemos guardar el número con su primer "1" en la primera de las ocho posiciones del byte siguiente. Así, si la representación binaria fuera 1∅1∅1∅1∅1∅, esto se guardaría en los dos bytes que hay a continuación del byte del exponente como 1∅1∅1∅1∅1∅∅∅∅∅∅∅ y en el byte del exponente se reflejaría el hecho de que tan sólo son significativos los diez primeros dígitos.

Nuestra tarea final es transformar el primer "1" de nuestro número binario en un cero, para indicar al Spectrum que se trata de un número positivo. El "1" se restablecerá cuando se evalúe el número.

El número necesario se crea en la línea 9994, sin ninguna referencia evidente a los números binarios. El exponente y el destino, una vez traducido, se colocan en los tres primeros bytes de la representación de cinco bytes. A continuación la rutina continúa con la renumeración del resto del programa.

Comprobación de 5.4

La primera regla que debe observarse cuando se comprueba una rutina como ésta es grabarla antes de intentar utilizarla. Los programas que cambian valores de la memoria del Spectrum pueden ser incorrectos y hacer que se pierda la máquina. Esto no tiene ninguna importancia para el Spectrum, pero puede perder el programa. Una vez guardada la rutina de renumeración, tendrá que darle algo sobre lo que trabajar; esto tan sólo tiene que ser un programa de tres líneas como un GOTO ∅∅∅5, 2 REM, 5 REM. Ahora cambie la segunda sentencia en la línea 9963 por IF LINEA=9958 THEN STOP. Ejecute el

programa, empezando en 9959. Cuando el programa se detenga, visualice T$. Deberá ser algo como: — 23759ØØØ5.

Si la comprobación es satisfactoria, corrija la línea 9963 y ejecute de nuevo el programa desde 9959. Esta vez deberá renumerar las líneas. Visualice BYTE 1 y BYTE 2. Deberán ser 138 y 6528Ø respectivamente. Si el programa no funciona hasta este punto, intente insertar algunas sentencias de STOP apropiadas en varios puntos para comprobar que el programa va siguiendo el proceso descrito anteriormente.

Resumen

Si puede escribir una rutina como ésta en código máquina, hágalo. Será cien veces más rápida. Pero no se quede con la idea de que el manipular cosas como la representación en cinco bytes de Sinclair no puede hacerse en BASIC. El intentarlo con código máquina puede dar como resultado algunos programas que no hagan casi nada aunque lo hagan muy rápido. Por lo tanto asegúrese de que realmente quiere dejar la comodidad del BASIC para conseguir sus objetivos. Quien sabe, quizá dentro de diez años la única diferencia entre un programa BASIC y una versión en código máquina sea una milésima de segundo.

Posibles mejoras

1) A mí me gustan los programas numerados empezando en 1ØØØ y de diez en diez, quizás a usted no. ¿Por qué no añadir un sencillo mecanismo que permita al usuario especificar la base y el espaciado entre líneas?
2) Vea hasta dónde puede llegar para simplificar la rutina, utilizando funciones definidas.
3) El Spectrum puede tratar GOTOs que apuntan a una línea no existente. Esta rutina no renumerará GOTOs. Añada una función al programa que recorra los números de línea para comprobar que exista un destino para cada GOTO o GOSUB.

5.5 Archivo II

Con Archivo II volvemos al tema con el que iniciamos este libro, es decir, el archivado flexible de grandes cantidades de información. En el siguiente programa veremos una nueva aplicación de los mé-

todos discutidos al principio, en el capítulo 1, y veremos de nuevo la potencia de nuestro planteo modular. La numeración del programa es principalmente la del programa original Archivo y, a diferencia de otros programas de este libro, las modificaciones realizadas no han sido renumeradas. Esto se ha hecho para que usted pueda identificar exactamente dónde se han realizado cambios o se han añadido cosas con respecto al programa original y por lo tanto, excepto en el caso de los comentarios sobre los cambios realizados, podrá referirse a los comentarios de la versión original.

La nueva aplicación para la que se ha realizado este programa es la de una base de datos. Una base de datos, en su forma más sencilla, es una colección de información, preferiblemente que incorpore formas de añadir datos al stock actual y formas de descubrir lo que hay allí guardado. A diferencia de la clara y pulida estructura de los archivos creados por el programa original Archivo, la base de datos necesita poder aceptar información cuya estructura no puede preverse tan fácilmente. La estructura puede variar mucho de un registro a otro. La base de datos suele utilizar un método más sofisticado de búsqueda, permitiendo al usuario especificar un cierto número de características que debe reunir el registro buscado. Aunque éstas son especificaciones bastante duras, debido a la potencia del planteo modular a la hora de escribir programas, este programa necesitó unas tres horas para ser escrito, partiendo de la idea original, hasta llegar a la versión terminada. A continuación los módulos están únicamente comentados con respecto a lo que difieren de forma significativa del módulo equivalente de Archivo. Para entrar el programa aconsejamos al lector que cargue primeramente el programa Archivo original y trabaje sobre éste, modificándolo de acuerdo con el listado de este programa.

MODULO 5.5.1

```
1000> PAPER 7: CLS : BORDER 7: I
NK 6: PAPER 0: PRINT PAPER 2;"
         ARCHIVO              "
1010 PRINT '"FUNCIONES DISPONIBL
ES:"
1020 PRINT '"         1)CREAR NUEVO
ARCHIVO"
1030 PRINT '"         2)ENTRAR INFOR
MACION"
1040 PRINT '"         3)BUSCAR/VISUA
LIZAR/CAMBIAR"
1045 PRINT '"         4)NOMBRES DE E
TIQUETAS"
1050 PRINT '"         5)FINALIZAR"
1060 PRINT ''"ENTRE POR FAVOR EL
 QUE REQUIERA."
1070 INPUT Z$
1080 CLS
1090 IF Z$="1" THEN GO SUB 1210
```

```
1100 IF Z$="2" THEN GO SUB 1440
1110 IF Z$="3" THEN GO SUB 2180
1115 IF Z$="4" THEN GO SUB 3400
1120 IF Z$="5" THEN GO SUB 1150
1130 CLS
1140 GO TO 1000
1150 PRINT AT 10,3; INK 7; PAPER
 2;"SISTEMA DE ARCHIVO CERRADO"
1160 BEEP 2,2
1180 INPUT "Ha entrado nueva inf
ormacion que desee guardar? (S/N
)";Q$: IF Q$="N" THEN STOP
1190 SAVE "ARCHIVO": PRINT '"Reb
obine la cinta, luego pulse cua
lquier tecla para VERIFICAR": PA
USE 0: VERIFY "ARCHIVO": STOP
```

El módulo está modificado en las líneas 1∅45 y 1115 para tener en cuenta una nueva función, que se explicará más adelante.

MODULO 5.5.2

```
1200>REM ***********************
1210 REM INICIALIZAR
1220 REM ***********************
1230 DIM A$(50,10)
1350 DIM B$(28000)
1360 LET B$(1 TO 4)=CHR$ 2+CHR$
0+CHR$ 2+CHR$ 255
1370 DEF FN A()=256*CODE Y$(2*S-
1)+CODE Y$(2*S)
1380 DEF FN A$()=B$(C TO C+CODE
B$(C)-1)
1390 LET P=5
1400 LET Y$=CHR$ 0+CHR$ 1+CHR$ 0
+CHR$ 3
1410 LET N=2
1420 RETURN
```

Una gran parte de este módulo se ha borrado para tener en cuenta el hecho de que no existirá una estructura regular para los nombres de los elementos de cada registro. Sin embargo existirá la posibilidad de asociar nombres a los elementos y estos nombres se guardarán en la tabla A$

MODULO 5.5.3

```
2750>REM ***********************
2760 REM SUBRUTINAS FUNCIONALES
2770 REM ***********************
2780 INPUT Q$
2785 IF LEN Q$>1 THEN RANDOMIZE
: IF Q$(LEN Q$-1)="#" THEN LET Q
$=Q$( TO LEN Q$-1)+CHR$ (VAL Q$(
LEN Q$))
2787 IF LEN Q$>2 THEN RANDOMIZE
: IF Q$(LEN Q$-2)="#" THEN LET Q
$=Q$( TO LEN Q$-2)+CHR$ (VAL Q$(
LEN Q$-1 TO ))
```

```
2790 LET Q$=CHR$ (LEN Q$+1)+Q$
2800 RETURN
2810 PRINT A$(I,2 TO CODE A$(I,1
));":";
2820 RETURN
2850 IF FN A$() (2 TO )="*" THEN
RETURN
2860 IF FN A$() (LEN FN A$()-1)="
#" THEN PRINT A$(CODE FN A$() (LE
N FN A$()));":";
2870 PRINT FN A$() (2 TO LEN FN A
$()-2*(FN A$() (LEN FN A$()-1)="#
"))
2880 LET C=C+CODE B$(C)
2890 GO TO 2850
```

La pequeña rutina de la línea 278∅ está modificada para que pueda reconocer números de la etiqueta precedidos por # y convertir el número que va a continuación en un único carácter que corresponda a este código. La rutina de la línea 285∅ está cambiada para que visualice elementos hasta que detecte el simbolo "*" que indica el final. Las etiquetas apropiadas se visualizan en la línea 286∅, si está presente el símbolo "#".

MODULO 5.5.4

```
1430>REM ************************
1440 REM ENTRADA NORMAL
1450 REM ************************
1460 LET R$=""
1470 PRINT PAPER 2;"           EN
TRADAS                 "
1480 PRINT '"COMANDOS DISPONIBLE
S:"
1490 PRINT '">ENTRAR ELEMENTO ES
PECIFICADO"''">""ZZZ"" PARA SALI
R"
1500 PRINT "********************
************"
1510 PRINT "LONGITUD ARCHIVO:";P
-1;"/";LEN B$
1540 GO SUB 2780
1550 IF Q$(LEN Q$-1)="#" THEN PR
INT A$(CODE Q$(LEN Q$));":";
1580 PRINT Q$(2 TO LEN Q$-2*(Q$(
LEN Q$-1)="#"))
1590 IF LEN Q$>=4 THEN RANDOMIZE
 : IF Q$(2 TO 4)="ZZZ" THEN RETU
RN
1600 LET R$=R$+Q$
1610 IF Q$(2)<>"*" THEN GO TO 15
20
1620 CLS
1630 GO SUB 1660
1640 GO TO 1440
```

En lugar de pedir un número especificado de elementos, ahora este módulo acepta elementos hasta que se encuentra con uno que consiste en un "*", el cual es interpretado como indicador del final del

registro. Las líneas 155∅ y 158∅ se refieren al método de etiquetado del elemento. Para hacerlo, el usuario añade "#" y a continuación un número comprendido entre 1 y 5∅ al final del elemento. Esto se interpreta como indicativo de un número de un elemento de la tabla A$ y la línea correspondiente de A$ se visualiza mediante la línea 155∅ si se detecta el #. La línea 158∅ asegura que no se visualice el número.

MODULO 5.5.5

```
1650>REM ************************
1660 REM PONER DATOS EN ARCHIVO
1670 REM ************************
1680 IF P+LEN R$-1<LEN B$ THEN G
O TO 1730
1690 PRINT AT 14,10;"ARCHIVO LLE
NO"
1700 PRINT ''"Pulsar cualquier t
ecla para"'"continuar"
1710 PAUSE 0
1720 RETURN
1730 LET POTEN=INT (LN (N-1)/LN
2)
1740 LET S=2↑POTEN
1750 LET T$=R$(2 TO CODE R$(1))
1760 FOR K=POTEN-1 TO 0 STEP -1
1770 LET C=FN A()
1780 LET U$=FN A$() (2 TO )
1790 LET S=S+(2↑K)*(T$>U$)-(2↑K)
*(T$<U$)
1810 IF S>N-1 THEN LET S=N-1
1820 IF S<2 THEN LET S=2
1830 NEXT K
1840 LET C=FN A()
1850 LET U$=FN A$() (2 TO )
1860 IF T$<U$ THEN LET S=S-1
1870 LET B$(P TO P+LEN R$-1)=R$
1880 LET N=N+1
1890 LET Y$=Y$(1 TO 2*S)+CHR$ IN
T (P/256)+CHR$ (P-256*INT (P/256
))+Y$(2*(S+1)-1 TO )
1900 LET P=P+LEN R$
1910 RETURN
```

Aquí no se ha hecho ningún cambio.

MODULO 5.5.6

```
2170>REM ************************
2180 REM BUSQUEDA
2190 REM ************************
2200 LET S=2
2210 PRINT PAPER 2;"
BUSQUEDA                        "
2220 PRINT ''"COMANDOS DISPONIBL
ES:"
2230 PRINT ">ENTRAR ELEMENTO PAR
A LA"'" BUSQUEDA NORMAL"'">PRECE
DER CON ""SSS"" PARA LA"'" BUSQU
EDA ESPECIAL"'">PRECEDER CON ""I
```

```
II"" PARA BUSCAR"'" POR EL PRIME
R CARACTER DEL"'" REGISTRO"'">""
ENTER"" PARA OBTENER EL PRIMER"'
" ELEMENTO DEL ARCHIVO"
2235 PRINT ">""MMMM"" PARA BUSQU
EDA MULTIPLE"
2240 PRINT "********************
***********"
2250 PRINT ''"ENTRAR ELEMENTO A
BUSCAR:";
2260 GO SUB 2780
2270 PRINT Q$(2 TO )
2280 LET S$=Q$
2290 IF LEN S$=1 THEN GO TO 2510
2300 LET C=FN A()
2310 IF LEN S$<5 THEN GO TO 2430
2311 IF S$(2 TO 4)<>"MMM" THEN G
O TO 2320
2312 PRINT "NUMERO DE ELEMENTOS
DE BUSQUEDA? ";: INPUT BUSQUEDA:
 PRINT BUSQUEDA: DIM M$(BUSQUEDA
,10): FOR K=1 TO BUSQUEDA: PRINT
 "ELEMENTO DE BUSQUEDA ";K;": ";
: GO SUB 2780: PRINT Q$(2 TO ):
LET M$(K)=Q$: NEXT K
2313>LET S1=S
2314 FOR K=1 TO BUSQUEDA: LET S$
="MMMM"+M$(K,2 TO CODE M$(K,1)):
 GO SUB 2940: IF C4=0 THEN RETUR
N
2316 IF S=S1 THEN NEXT K: GO TO
2510
2318 GO TO 2313
2320 IF S$(2 TO 4)<>"III" THEN G
O TO 2390
2330 FOR I=S TO N
2340 LET S=I
2350 LET C=FN A()
2360 IF B$(C+1)=S$(5) THEN GO TO
 2510
2370 NEXT I
2380 RETURN
2390 IF S$(2 TO 4)<>"SSS" THEN G
O TO 2430
2400 GO SUB 2920
2410 IF C4=1 THEN GO TO 2510
2420 RETURN
2430 IF FN A$()=S$ THEN GO TO 25
10
2450 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2460 LET C=C+CODE B$(C)
2470 IF B$(C-1)<>"*" THEN GO TO
2430
2480 LET S=S+1
2490 LET C=FN A()
2500 GO TO 2430
2510 LET C=FN A()
2520 LET C4=0
2530 IF FN A$()=CHR$ 2+CHR$ 255
THEN RETURN
2540 CLS
2550 PRINT "REGISTRO ";S-1;":-"
2560 GO SUB 2850
2570 LET S=S+1
2580 PRINT AT 14,0; PAPER 2;"
         BUSQUEDA              "
2590 PRINT "COMANDOS DISPONIBLES
:"
2600 PRINT ">""ENTER"" PARA VISU
ALIZAR EL"'" SIGUIENTE ELEMENTO"
```

```
'">""ZZZ"" PARA ABANDONAR LA FUN
CION"'">""AAA"" PARA MODIFICAR"'
">""CCC"" PARA CONTINUAR LA"'" B
USQUEDA"
2610 INPUT P$
2620 CLS
2625 IF LEN S$>=4 THEN RANDOMIZE
 : IF P$="CCC" AND S$(2 TO 4)="M
MM" THEN GO TO 2313
2630 IF P$="CCC" THEN GO TO 2300
2640 IF P$="" THEN GO TO 2510
2650 IF P$<>"AAA" THEN GO TO 271
0
2660 LET C=FN A()
2670 CLS
2680 GO SUB 1930
2710 IF P$="ZZZ" THEN RETURN
2720 IF P$="AAA" THEN RETURN
2730 CLS
2740 GO TO 2260
```

Entre las líneas 2311 y 2318 se ha añadido una pequeña rutina que acepta un cierto número de elementos que deben guardarse, y después llama al módulo de búsqueda especial para que busque sucesivamente cada uno de los elementos. Si todas las combinaciones de caracteres especificadas se encuentran dentro de un registro, entonces se visualiza este registro. Obsérvese también la pequeña modificación del menú en 2235 y de la orden de continuación en 2625.

Comprobación del módulo 5.5.6

Ahora deberá ser capaz de visualizar elementos, aunque todavía no puede utilizar la búsqueda especial ni la búsqueda múltiple. Si especifica números de etiqueta en la entrada, el elemento será visualizado mediante 1∅ caracteres y precedido por un ":".

MODULO 5.5.7

```
2910>REM ***********************
2920 REM BUSQUEDA ESPECIAL
2930 REM ***********************
2940 LET C4=0
2950 FOR H=5 TO N-1
2960 LET S=H
2970 LET C=FN A()
2980 LET C1=C
3000 LET C1=C1+CODE B$(C1)
3010 IF B$(C1-1)<>"*" THEN GO TO
 3000
3020 FOR J=C+1 TO C1-LEN S$+5
3030 IF B$(J TO J+LEN S$-5)<>S$(
5 TO ) THEN GO TO 3060
3040 LET C4=1
3050 RETURN
3060 NEXT J
3070 NEXT H
3080 LET C4=0
3090 RETURN
```

Se ha realizado un pequeño cambio en este módulo para tener en cuenta la falta de estructura regular de un registro.

Comprobación del módulo 5.5.7

Ahora ya podrá realizar búsquedas para combinaciones de caracteres —elemento precedido por SSS—. También podrá entrar en el módulo de búsqueda múltiple.

MODULO 5.5.8

```
1920>REM ***********************
1930 REM CAMBIAR REGISTRO
1940 REM ***********************
1950 LET S=S-1
1960 LET C=FN A()
1970 LET R$=""
1980 PRINT "REGISTRO ";S-1;": -"
2000 IF FN A$() (2)="*" THEN LET
R$=R$+CHR$ 2+"*": GO SUB 3130: G
O SUB 1660: RETURN
2010 PRINT FN A$() (2 TO LEN FN A
$()-1);
2015 IF FN A$() (LEN FN A$()-1)="
#" THEN PRINT CODE (FN A$() (LEN
FN A$()))
2017 IF FN A$() (LEN FN A$()-1)<>
"#" THEN PRINT FN A$() (LEN FN A$
())
2020 PRINT AT 16,0; PAPER 2;"
       MODIFICAR                 "
2030 PRINT "COMANDOS DISPONIBLES
:"
2040 PRINT ">""ENTER"" NO MODIFI
CAR"'">""ZZZ"" ELIMINA TODO EL R
EGISTRO"'">ELEMENTO CAMBIADO"'">
NUEVO ELEMENTO ACABADO EN ""*"""
2050 GO SUB 2780
2060 IF LEN Q$=1 OR Q$(LEN Q$)="
*" THEN LET R$=R$+FN A$()
2070 LET C=C+CODE B$(C)
2080 CLS
2090 IF LEN Q$=1 THEN GO TO 2000
2100 IF Q$(2 TO )="ZZZ" THEN GO
TO 3130: RETURN
2105 IF Q$(LEN Q$)="*" THEN LET
Q$=Q$(2 TO LEN Q$-1): GO SUB 278
5
2110 LET R$=R$+Q$
2120 GO TO 2000
2130 GO SUB 3130
2140 IF Q$(2 TO )="ZZZ" THEN RET
URN
2150 GO SUB 1660
2160 RETURN
```

Líneas 2Ø15-2Ø17. Reflejan la necesidad de convertir un número de etiqueta, si es que está presente, partiendo del carácter de código correspondiente, en el número que representa. Si no hay número de

etiqueta, se visualizan los dos últimos caracteres del elemento. El menú y las líneas subsiguientes están modificadas para tener en cuenta a los elementos con un elemento terminal "*" —cuando se entra uno de éstos, se guarda en el registro, a continuación del elemento que se está visualizando en la parte superior de la pantalla. Esto significa que no puede insertarse un nuevo elemento al principio de un registro.

MODULO 5.5.9

```
3100>REM ***********************
3110 REM DESPLAZAR ARCHIVO
3120 REM ***********************
3130 LET C=FN A()
3140 LET DESPL=1000
3150 LET C1=C
3160 LET C3=C
3180 LET C1=C1+CODE B$(C1)
3190 IF B$(C1-1)<>"*" THEN GO TO
 3180
3200 LET C2=C1-C
3210 FOR I=C1 TO LEN B$-1 STEP D
ESPL
3220 IF LEN B$-I+1<DESPL THEN LE
T DESPL=LEN B$-I+1
3230 LET S$=B$(I TO I+DESPL-1)
3240 LET B$(C TO C+DESPL-1)=S$
3250 LET C=C+DESPL
3260 NEXT I
3270 LET Y$=Y$(1 TO 2*(S-1))+Y$(
2*(S+1)-1 TO )
3280 FOR I=1 TO N-1
3290 LET S=I
3300 LET C=FN A()
3310 IF C<=C3 THEN GO TO 3350
3320 LET C=C-C2
3330 LET Y$(2*I-1)=CHR$ INT (C/2
56)
3340 LET Y$(2*I)=CHR$ (C-256*INT
 (C/256))
3350 NEXT I
3360 LET P=P-C2
3370 LET N=N-1
3380 RETURN
```

Se han realizado cambios en la línea 319Ø y se ha borrado la línea 317Ø.

Comprobación del módulo 5.5.9

La función de modificación ahora debe ser totalmente operativa, permitiendo borrados, alteraciones e inserciones.

```
3400>REM ***********************
3410 REM ETIQUETAS DE ELEMENTO
3420 REM ***********************
3430 FOR I=1 TO 50 STEP 10
3440 CLS : FOR J=I TO I+9
3450 PRINT J;") ";A$(J)
3460 NEXT J
3470 PRINT '''"ORDENES:"
3480 PRINT '">ZZZ=SALIR"
3490 PRINT ">III=ELEMENTO"
3495 PRINT ">NNN=PAGINA SIGUIENT
E"
3500 INPUT Q$: IF Q$="ZZZ" THEN
RETURN
3505 IF Q$="NNN" THEN CLS : NEXT
 I: RETURN
3510 IF Q$="III" THEN INPUT "NUM
ERO? ";TIPO: INPUT "ETIQUETA DE
ELEMENTO? ";Q$: LET A$(TIPO)=Q$:
 CLS : GO TO 3440
3520 GO TO 3500
```

Este pequeño módulo permite al usuario entrar etiquetas para los elementos situados en las posiciones especificadas de la tabla A$: no se ha realizado una previsión específica para el borrado, que no obstante puede conseguirse fácilmente especificando la línea correspondiente y entrando un texto vacío como su nuevo contenido.

Comprobación del módulo 5.5.10

Si previamente se han especificado números de etiqueta para algunos de los elementos, ahora podrá suministrar algún contenido para estas etiquetas y verlas visualizadas delante de los elementos correspondientes mediante el módulo de visualización.

Resumen

Sería una locura sugerir que un programa como éste es tan útil y tan rápido como las bases de datos en código máquina, capaces de realizar una búsqueda completa en segundos. No obstante, el programa funciona y tiene características superiores a algunos programas de bases de datos baratos que se venden para microordenadores personales. La búsqueda múltiple es un lujo que consume tiempo y que probablemente no deseará utilizar muy a menudo sobre archivos grandes. Quizá, debido a necesidades específicas propias, utilice esta versión de Archivo con más frecuencia que la original.

La lección realmente interesante que hay que sacar de esto es que una nueva aplicación de cierta importancia se ha convertido en

algo muy sencillo mediante el planteo modular que nos ha permitido identificar claramente las áreas que necesitaban modificación.

Posibles mejoras

1) Una vez advertidos con respecto a los problemas del código máquina hay que admitir que con una aplicación como ésta estamos en las fronteras de lo que puede conseguirse para que sea de utilidad con el BASIC en una máquina con la velocidad del Spectrum. Mis propias versiones de Archivo para el ZX81 incorporan pequeñas rutinas en código máquina para realizar las búsquedas que consumen más tiempo. Si las aplicaciones de bases de datos van a ser cruciales para la utilización que va a darle al Spectrum, entonces ésta será la dirección que tendrá que seguir.

5.6 Mecanografía

No todos los programas útiles tienen que tener cientos de líneas. Este le dará un buen empujón a la hora de mejorar su mecanografía si lo utiliza adecuadamente. Es corto porque, al igual que con el programa de Geografía que vimos anteriormente, se deja para el usuario la entrada y borrado del material con el que trabajará el programa, en forma de sentencias DATA. Por lo tanto, no aparecen los largos módulos para realizar la entrada de los datos y para el borrado de los mismos. El programa sirve como recordatorio de la constante necesidad de examinar lo que se intenta realizar con el Spectrum y plantear la pregunta: ¿Tiene que ser tan sofisticado? A veces la respuesta es claramente sí, como en el caso de un programa que sea realmente de propósito general. Muchas veces es evidente que la estructura de un programa es tal que es muy poco probable que sea de utilidad para cualquier otro propósito distinto para el que fue diseñado.

MODULO 5.6.1

```
1270>REM ************************
1280 REM CARACTERES USR
1290 REM ************************
1300 RESTORE : FOR I=0 TO 63: RE
AD BYTE: POKE USR "A"+I,BYTE: NE
XT I: RETURN
1310 DATA 0,0,0,0,0,7,4,4
1320 DATA 0,0,0,0,0,255,0,0
1330 DATA 0,0,0,0,0,224,32,32
1340 DATA 4,4,4,4,4,4,4,4
```

```
1350 DATA 32,32,32,32,32,32,32,3
2
1360 DATA 4,4,7,0,0,0,0,0
1370 DATA 0,0,255,0,0,0,0,0
1380 DATA 32,32,224,0,0,0,0,0
1390 RETURN
```

Este módulo rellena los primeros ocho caracteres del área de gráficos definidos por el usuario con los caracteres necesarios para dibujar un pequeño cuadrado alrededor de una letra. Estos cuadrados se utilizarán para representar las teclas sobre la pantalla. El método utilizado es similar al utilizado en el juego Cacería.

MODULO 5.6.2

```
1000>REM ************************
1010 REM VISUALIZAR TECLADO
1020 REM ************************
1030 INK 0: PAPER 6: CLS : DIM O
$(32): GO SUB 1270
1040 CLS : PRINT "
                   □ □ □ □ [
] □ □ □ □ □ "
1050 PRINT "
              □ □ □ □ □ □ [
] □ □ □ "
1060 PRINT "
               □ □ □ □ □ □
□ □ □ □ "
1070 PRINT "
              □ □ □ □ □ □ [
] □ □ □ "
1080 PRINT OVER 1;AT 1,0;" 1  2
 3  4  5  6  7  8  9  0"
1090 PRINT OVER 1;AT 4,0;"  Q  W
  E  R  T  Y  U  I  O  P"
1100 PRINT OVER 1;AT 7,0;"   A
S  D  F  G  H  J  K  L  ■"
1110 PRINT OVER 1;AT 10,0;" ■
Z  X  C  V  B  N  M  ■"
```

Este módulo dibuja una copia rudimentaria del teclado de una máquina de escribir, en la parte superior de la pantalla, utilizando los caracteres definidos en el módulo anterior.

Comentario

Líneas 1Ø4Ø-1Ø7Ø. Estas cadenas de letras son entradas en modo gráfico y de hecho crearán los perfiles de los cuadrados de las teclas.

Líneas 1Ø8Ø-111Ø. Cada tecla es etiquetada con la letra principal

o el número que contiene. La tecla de espacio se deja en blanco y las teclas de CAPS SHIFT, SYMBOL SHIFT y ENTER se etiquetan con sus iniciales en visualización invertida.

Comprobación del módulo 5.6.2

Coloque temporalmente un STOP al final del módulo y entonces podrá visualizar un teclado reconocible sobre la pantalla.

MODULO 5.6.3

```
1400 >REM ***********************
1410 REM INFORMACION PRUEBAS
1420 REM ***********************
1430 DATA "ESTO ES UNA PRUEBA EN
 SPECTRUM"
1440 DATA "TECLEE EXACTAMENTE LO
 QUE VEA"
1450 DATA "NO MIRE AL TECLADO"
1460 DATA "STOP"
```

Este módulo se utilizará para guardar los datos para los tests de mecanografía. Los textos individuales no deben ser más largos de 32 caracteres. El módulo debe terminarse con una línea de DATA que contenga un 19∅ STOP, que se utiliza como una señal para volver a colocar el puntero de datos en el módulo siguiente.

MODULO 5.6.4

```
1120 >REM ***********************
1130 REM ACEPTAR ENTRADA
1140 REM ***********************
1150 LET TOTAL=0: LET BIEN=0
1160 READ A$: IF A$="STOP" THEN
RESTORE 1430: READ A$
1170 PRINT INK 1;AT 16,0;O$;AT 1
6,0;A$'O$: PRINT AT 17,0;
1180 FOR I=1 TO LEN A$
1190 LET T$=INKEY$: IF T$<>"" TH
EN GO TO 1210
1200 GO TO 1190
1210 LET TOTAL=TOTAL+1: BEEP .1,
30: IF T$<>A$(I) THEN PRINT PAPE
R 5;T$;CHR$ 8;: GO TO 1190
1220 LET BIEN=BIEN+1
1230 PRINT BRIGHT 1;T$;
1240 NEXT I: PRINT AT 21,25;INT
(BIEN/TOTAL*100);"%  "
1250 INPUT "MAS? (S/N) ";Q$: IF
Q$<>"N" THEN GO TO 1160
1260 STOP
```

Este módulo visualiza un texto que debe teclearse e invita al usuario a que lo copie sobre la pantalla, sin mirar al teclado real. Las

pulsaciones incorrectas no son aceptadas pero se mantiene un registro de los errores, que será visualizado al final de cada texto.

Comentario

Línea 116∅. Esta línea recorre los datos del módulo 3 y vuelve a posicionar el puntero al principio de los datos si se encuentra con un STOP.

Líneas 118∅-124∅. Este bucle acepta la entrada —la variable del bucle se incrementa únicamente con las pulsaciones correctas.

Línea 121∅. Las pulsaciones incorrectas se visualizan para que pueda verlas el usuario, diferenciadas por un fondo de color cian. Obsérvese la utilización del carácter de control, CHR$8, para mover la posición de visualización una posición hacia atrás cuando se ha visualizado un carácter incorrecto, por lo que la posición de visualización permanece en el mismo lugar cuando se ha entrado un carácter incorrecto.

Línea 123∅. Las pulsaciones correctas se añaden al texto que se está tecleando. La característica BRIGHT se coloca para que el usuario pueda comprobar que se han entrado los espacios.

Comprobación del módulo 5.6.4

Si ha entrado algunos datos en el módulo 3, ahora podrá ejecutar el programa y ver visualizado el teclado. El primer texto del área de datos deberá visualizarse debajo del mismo y podrá intentar reescribirlo.

Resumen

La utilidad de este programa depende de si está dispuesto a tomárselo en serio. Utilizando junto con un manual para enseñar mecanografía que haya conseguido de una biblioteca y del que puedan obtenerse el tipo correcto de ejercicios, puede ser una herramienta muy efectiva. Para empezar probablemente encuentre más fácil limitarse a las letras mayúsculas, o minúsculas, sin mezclarlas con signos de puntuación o símbolos. Cuando esté preparado, el programa tratará con caracteres mezclados y además los símbolos, con la tecla SYMBOL SHIFT.

Posibles mejoras

1) Utilizando el método señalado en la página 131 del manual del Spectrum, añada una función de temporización a la entrada, después modifique el programa para que pueda visualizar una serie de textos que lleguen a, por ejemplo, 1ØØØ caracteres. Anote el tiempo desde que empieza a entrar hasta que temine el test y tendrá, considerándolo junto con el tanto por ciento de pulsaciones, un buen indicador de sus progresos.
2) Modifique la visualización de forma que tan sólo visualice el teclado cuando cometa un error —quizás en el caso de más de un error en el mismo carácter— y lo elimine cuando el carácter se haya entrado correctamente.

6. Y finalmente un poco de diversión. Juegos para el Spectrum

Quizá se haya dado cuenta con el contenido de este libro que yo no considero los juegos como el no va más de los ordenadores domésticos. Sospecho que, muchas veces, los juegos suelen ser el punto donde se encallan los que han descubierto la fascinación de los ordenadores, pero que todavía no han explorado las formas en que la potencia del microordenador puede mejorar la vida diaria.

Sin embargo, los juegos tienen su lugar, en función de los propios juegos. El colocar un juego poco interesante en un micro no hace que éste sea mejor y los caracteres definidos por el usuario no cambian el hecho de que, en BASIC, el Spectrum es demasiado lento para ejecutar el tipo de juegos de invasores con un grado real de satisfacción.

Los micros son excelentes con juegos que requieren pensar. A continuación encontrará dos juegos que dependen más de pensar que de la velocidad de reacción. El primero de ellos, Misil, a primera vista parece una copia directa del tipo de juegos de caza de extraterrestres, pero requiere la realización de ciertos cálculos por parte del jugador. A continuación viene Cacería, un interesante juego en el que hay que cazar una presa invisible. Finalmente, he incluido una sencilla rutina de clasificación que debe demostrar su utilidad para los adictos a los juegos de rompecabezas con palabras y otros tipos de juegos en los que debe utilizarse una clasificación numérica o de textos.

6.1 Misil

No puede ganar en este juego. Al final, los terribles extraterrestres llegarán a la parte inferior de la pantalla y le destruirán. Su función es destruir tantos extraterrestres como pueda antes de que le destruyan. Sus armas consisten en misiles capaces de transportar cabezas explosivas de distinta fuerza, que se disparan desde seis bases de misiles. Las naves extraterrestres, representadas por números del Ø al 9, descienden aleatoriamente, una línea cada vez, desde la parte superior de la pantalla. Si llegan a la parte inferior de la pantalla, se resta su fuerza del stock que tiene de cabezas explosivas. A cada

turno se invita al jugador a que especifique la base desde donde va a disparar, el ángulo que seguirá la trayectoria del misil —desde la posición recta hacia arriba, hasta 45° a la derecha, la altura a la que el misil explotará— siempre que el misil no choque antes con algo, y la potencia de la cabeza explosiva —que se entra como una potencia de 2 hasta un máximo de 4—, es decir, 16.

Cuando el misil explota, si ha alcanzado a una nave extraterrestre, destruirá la potencia de esta nave en un valor igual al doble de su propia potencia. A una nave extraterrestre que esté en el borde del cuadrado en que explota el misil, se le reducirá su fuerza en un valor equivalente a la potencia del misil. Las naves que estén a una distancia de un cuadrado del lugar donde explote el misil, perderán una fuerza equivalente a la mitad de la potencia del misil, y así sucesivamente. Cuando la fuerza de una nave extraterrestre descienda por debajo de cero, desaparecerá. A cada movimiento se generan más extraterrestres.

Cada cinco movimientos, una nave de suministro, representada por un "*", empieza a descender desde la parte superior de la pantalla entre las naves extraterrestres. Si consigue aterrizar, aumentará su stock de cabezas explosivas en 1ØØ unidades. Pero si en cualquier punto llega a estar dentro del círculo de destrucción creado por un misil, se perderá. La puntuación consiste en la cantidad total de daños infringidos por el jugador a las naves extraterrestres y el juego termina cuando se termina el stock de cabezas explosivas. Una complicación final consiste en que el jugador tiene una cantidad de tiempo limitada en la cual realizar cualquier entrada. Se pueden establecer niveles variables de dificultad. Si se excede este tiempo o se realiza una entrada inapropiada, los extraterrestres descenderán una línea sin estorbos.

MODULO 6.1.1

```
1000>REM ************************
1010 REM VARIABLES
1020 REM ************************
1030 DIM A$(608): DIM C$(4,6): D
IM C(4): DIM D(4,2): DIM O$(32)
1040 LET C$(1)="BASE?": LET C$(2
)="ANGL.?": LET C$(3)="DIST.?":
LET C$(4)="POTN."
1050 LET U$="": LET PUNT=0: LET
W=100: LET S=1: LET G=1
1060 LET D(1,1)=49: LET D(1,2)=5
4: LET D(2,1)=48: LET D(2,2)=57:
 LET D(3,1)=48: LET D(3,2)=57: L
ET D(4,1)=48: LET D(4,2)=54
1070 BORDER 0: PRINT INK 2; PAPE
R 6;AT 10,8;"ATAQUE DE MISILES"
1080 INPUT "Entre por favor el n
ivel de      dificultad desado (0
-10):";H: LET H=500-45*H
```

La utilización de estas variables será explicada en el curso del programa. El módulo también acepta el nivel de dificultad.

MODULO 6.1.2

```
1910 REM ************************
1920 REM GENERAR EXTRATERRESTRES
1930 REM ************************
1940 LET G=G+1: FOR I=1 TO INT (
RND*(G/5)+1)
1950 LET A=(RND*30+1)
1960 LET SH=INT (RND*10)
1970 LET A$(A)=CHR$ (48+SH)
1980 NEXT I
1990 IF INT ((G-5)/10)=(G-5)/10
THEN LET A$(A)="*"
2000 PRINT AT 0,0;A$: RETURN
```

Este módulo genera las naves extraterrestres.

Comentario

Línea 194∅. El número de extraterrestres generados aumenta con el número de turnos realizados hasta el momento.

Línea 195∅. La posición de una nave extraterrestre.

Línea 196∅-197∅. Se genera un carácter entre el ∅ y el 9.

Línea 199∅. Cada cinco movimientos se genera una nave de aprovisionamiento.

Línea 2∅∅∅. Se visualiza la tabla que contiene el tablero del juego.

Comprobación del módulo 6.1.2

Llamando a este módulo se debe obtener como resultado la visualización de números aleatorios en la parte superior de la pantalla.

MODULO 6.1.3

```
2010 REM ************************
2020 REM MOVER PANTALLA
2030 REM ************************
2040 LET A$(33 TO 608)=A$(1 TO 5
76): LET A$(1 TO 32)=O$: PRINT A
T 0,0;A$
```

El campo de juego se desplaza una línea hacia abajo de la pantalla.

Comprobación del módulo 6.1.3

Al llamar a este módulo después de este último, deberemos obtener como resultado el que los números generados aleatoriamente se muevan hacia abajo de la pantalla.

MODULO 6.1.4

```
1790>REM ************************
1800 REM CURSO DEL MISIL
1810 REM ************************
1820 LET M=577+5*(C(1)-1)
1830 FOR I=1 TO 18
1840 IF A$(INT M)<>" " THEN RETU
RN
1850 LET A$(INT M)="+"
1860 PRINT AT 0,0;A$
1870 IF I=C(3) THEN RETURN
1880 LET A$(INT M)=" "
1890 LET M=M-32+SIN C(2)
1900 NEXT I: RETURN
```

Este módulo mueve un "+", que representa el misil del jugador, hacia arriba de la pantalla.

Comentario

Línea 182∅. Se especifica el cuadrado de encima de la base.

Líneas 183∅-19∅∅. Si no hay ningún obstáculo y la altura máxima especificada no ha sido obtenida, el misil se mueve 32 espacios hacia arriba en la tabla unidimensional y después hacia atrás un número de espacios que representa el ángulo con el que se moverá el misil. El efecto neto producido es que el misil se mueve hacia arriba y hacia la derecha.

Comprobación del módulo 6.1.4

Si entra un número en radianes, que corresponda al ángulo del misil (C(2)) y una altura máxima (C(3)) y un número de base (C(1)), entonces al llamar a este módulo, deberá ver cómo se mueve el misil hacia arriba de la pantalla.

MODULO 6.1.5

```
1450>REM ************************
1460 REM CICLO DE ORDENES
1470 REM ************************
1480 INK 0: PRINT AT 20,0;O$;O$
```

```
1490 PRINT AT 21,0;"PUNT.:";PUNT
;,"CABEZAS :";W;"   "
1500 PRINT AT 20,0;
1510 FOR I=1 TO 4
1520 PRINT C$(I);
1530 FOR J=1 TO H+10
1540 LET T$=INKEY$: IF T$<>"" TH
EN GO TO 1570
1550 NEXT J
1560 PRINT AT 10,5;"DEFECTO EN "
;C$(I)
1565 FOR H=1 TO 200: NEXT H: LET
 U$="": GO SUB 2010: GO SUB 1910
: GO TO 1480
1570 BEEP .1,20: IF U$<>"" OR I<
>3 THEN GO TO 1600
1580 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1590 LET U$=T$: PAUSE 25: GO TO
1530
1600 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1620 LET T$=U$+T$: LET U$=""
1630 IF CODE T$>57 OR CODE T$<48
 THEN GO TO 1560
1640 IF CODE T$<D(I,1) OR CODE T
$>D(I,2) THEN GO TO 1560
1650 LET C(I)=VAL T$
1660 PRINT C(I);"■" AND I<>4;
1670 NEXT I
1680 LET I=I-1
1690 LET C(2)=C(2)*PI/36
1700 LET C(4)=2↑C(4)
1710 IF C(4)>W OR C(4)>16 THEN G
O TO 1560
1720 LET W=W-C(4)
1730 IF W>0 THEN GO TO 1770
1740 PRINT AT 10,0;"SE HAN ACABA
DO LAS CABEZAS EXPL."
1750 PRINT AT 11,6;"LA PUNTUACIO
N FUE ";PUNT
1760 STOP
1770 PRINT AT 21,0;"PUNT.:";PUNT
,"CABEZAS :";W;"   "
1780 RETURN
```

Este módulo acepta las órdenes del jugador e indica los fallos si los valores no están dentro de los límites correctos.

Comentario

Líneas 151∅-1565. Las cuatro órdenes, cuyos nombres están guardados en C$, se visualizan y después el bucle de la línea 153∅ da al jugador un corto período de tiempo en el cual debe pulsar una tecla. Si no se pulsa ninguna tecla, el programa considera un fallo y los extraterrestres se mueven hacia abajo.

Líneas 157∅-167∅. En función de qué orden se esté entrando, el módulo comprueba que la tecla pulsada caiga dentro de los límites que están guardados en la tabla D. Si se exceden los límites en cualquier orden, el módulo lo considera un fallo.

Línea 169∅. El ángulo entrado se multiplica por cinco, expresado en radianes.

Línea 171∅. Se coloca un límite superior de 16 para la potencia del misil, ya que valores más altos podrían generar caracteres inválidos en ciertos puntos del programa.

Comprobación del módulo 6.1.5

Ahora ya podrá entrar las cuatro órdenes, dentro de los límites establecidos por la tabla D —línea 1∅6∅. El programa deberá indicar un fallo si se exceden los límites o si no se realiza ninguna entrada.

MODULO 6.1.6

```
1180>REM ************************
1190 REM LETALIDAD
1200 REM ************************
1210 PRINT AT 20,0;O$
1220 PRINT AT 20,5;"*CALCULANDO
LETALIDAD*"
1230 LET M1=INT (M/32)
1240 LET M2=INT (M-32*M1)
1250 LET R=INT (.5+LN C(4)/LN 2)
1260 FOR J=M1-R TO M1+R
1270 PRINT BRIGHT 8;AT 0,0;A$
1280 PRINT BRIGHT 1;AT M1,M2-1;A
$(INT M)
1290 FOR K=M2-R TO M2+R
1300 IF 32*J+K<1 OR 32*J+H>608 T
HEN GO TO 1400
1310 IF J>=0 AND J<=21 AND K>0 A
ND K<33 THEN PRINT BRIGHT 1; OVE
R 1;AT J,K-1;" "
1320 IF A$(32*J+K)="+" THEN GO T
O 1400
1330 IF A$(32*J+K)=" " THEN GO T
O 1400
1340 LET D=ABS (K-M2)
1350 IF ABS (J-M1)>ABS (K-M2) TH
EN LET D=ABS (J-M1)
1360 LET E=INT (C(4)/(2↑(D-1)))
1370 LET A$(32*J+K)=CHR$ (CODE A
$(32*J+K)-E)
1380 IF CODE A$(32*J+K)<48 THEN
LET PUNT=PUNT+CODE A$(32*J+K)-48
: LET A$(32*J+K)=" "
1390 LET PUNT=PUNT+E
1400 NEXT K
1410 NEXT J
1420 IF A$(INT M)="+" THEN LET A
$(INT M)=" "
1430 INK 0: PAPER 5: PRINT AT 0,
0;A$
1440 RETURN
```

Este módulo calcula el efecto de una explosión sobre las naves extraterrestres.

Comentario

Líneas 123∅-124∅. Las coordenadas del misil están expresadas en términos bidimensionales.

Línea 125∅. R es el alcance del círculo de destrucción creado por la explosión.

Líneas 126∅-141∅. Con estos bucles, las posiciones de los caracteres que estén dentro de un rango de un radio R del punto de explosión del misil son examinados. Si la posición que se está examinando está vacía o contiene el propio misil, no se realiza ninguna acción. Si la posición contiene una de las naves extraterrestres, se decrementa su valor de acuerdo con la potencia del misil y la distancia a que se encuentra de la nave extraterrestre. Si el valor de la nave ha quedado reducido a menos de cero, se eliminará la nave correspondiente.

Comprobación del módulo 6.1.6

Llamando a este módulo después del módulo 4, deberá dar como resultado la visualización de un cuadrado sobreiluminado en la pantalla, en la última posición ocupada por el misil, siendo el tamaño del cuadrado dependiente de la potencia del misil.

MODULO 6.1.7

```
2050>REM ************************
2060 REM PENALIZACIONES Y BONOS
2070 REM ************************
2080 IF A$(577 TO 608)=O$ THEN R
ETURN
2090 FOR I=0 TO 31
2100 IF A$(577+I)="*" THEN LET W
=W+100
2110 IF CODE A$(577+I)>42 THEN L
ET W=W-CODE A$(577+I)+48
2120 NEXT I
2130 LET A$(577 TO 608)=O$: PRIN
T AT 18,0;O$
2140 PRINT AT 0,0;O$
2150 IF W>0 THEN RETURN
2160 PRINT AT 10,3;"EL ATAQUE EX
TRATERRESTRE TRIUNFA"
2170 PRINT AT 11,9;"ESTAS MUERTO
."
2180 PRINT AT 12,7;"LA PUNTUACIO
N FUE ";PUNT
2190 STOP
```

En el caso de que las naves extraterrestres alcancen la parte inferior de la pantalla, este módulo resta el valor de las naves del stock de cabezas explosivas del jugador.

MODULO 6.1.8

```
1090>REM ************************
1100 REM BUCLE PRINCIPAL
1110 REM ************************
1120 PRINT AT 19,0;"1####2####3#
###4####5####6######"
1130 GO SUB 1910
1140 INK 0: PAPER 5: PRINT AT 0,
0;A$
1150 PRINT AT 20,0;O$;O$
1170 GO SUB 1470: GO SUB 1820: G
O SUB 1180: GO SUB 2010: GO SUB
1910: GO TO 1150
```

Este módulo visualiza las bases en la parte inferior de la pantalla y después llama secuencialmente las distintas rutinas.

Comprobación del módulo 6.1.8

Ahora ya podrá realizar el juego.

Resumen

Este no es un juego rápido, pero vale la pena entrarlo por el sencillo motivo de que es apasionante en extremo. Cuando se juega con los niveles más altos de dificultad, el jugador está constantemente bajo presión a la hora de tomar decisiones y a cada nivel, una puntuación respetable (15ØØ o más) requiere rapidez de pensamiento y la realización de cálculos precisos.

El programa también proporciona un ejemplo de cómo puede utilizarse una tabla unidimensional para guardar un juego que ocupe la totalidad de la pantalla y manipularlo fácilmente. El método de dar una entrada temporizada en las líneas 153Ø-155Ø será de utilidad en una gran variedad de juegos en los que se desee poner al jugador bajo cierta presión.

Posibles mejoras

1) Este es un programa que pide a gritos que sea complementado con algunos gráficos definidos por el usuario. Podrían definirse los caracteres correspondientes a los misiles y a las distintas naves extraterrestres. En este caso, las líneas 137Ø y 138Ø necesitarían modificarse para que los caracteres procedieran de una base de CODE 144 en lugar de 48.

6.2 Cacería

Este es un juego enfurecedor que le hará dudar de su estado mental o de si el Spectrum funciona correctamente. Es un juego que se lleva a cabo sobre un tablero de 3Ø por 18, y en uno de los cuadrados se esconde la presa invisible. Su trabajo será seguir y descubrir a la presa, pero cada vez que haga un intento, la presa realizará un movimiento secreto. Sus únicas ventajas son que el movimiento de la presa es siempre el mismo dentro de la misma partida y que tendrá un indicador especial que le mostrará la dirección de la presa cada vez que haga un intento de adivinar su posición. Las instrucciones completas se visualizan en el propio juego.

Como táctica, es inteligente el tratar los dos componentes del movimiento de la presa —vertical y horizontal— por separado e intentar identificar uno de ellos antes de que nos distraigamos con el otro. Hay que tener mucho cuidado para asegurarnos de que se ha determinado la dirección correcta en que se mueve la presa. Es muy fácil trabajar sobre la suposición de que la presa se mueve sobre el tablero de izquierda a derecha —basándonos en dos o tres intentos y la clave correspondiente— mientras que en realidad todo el rato se estaba moviendo en la dirección opuesta.

MODULO 6.2.1

```
2180>REM ************************
2190 REM CARACTERES USR
2200 REM ************************
2210 RESTORE : FOR I=1 TO 8: FOR
 J=0 TO 7: READ B: POKE USR CHR$
 (143+I)+J,B: NEXT J: NEXT I: RE
TURN
2220 DATA BIN 00000000,BIN 01111
000,BIN 01100000,BIN 01010000,BI
N 01001000,BIN 00000100,BIN 0000
0010,BIN 00000001
2230 DATA BIN 00001000,BIN 00011
100,BIN 00101010,BIN 01001001,BI
N 00001000,BIN 00001000,BIN 0000
1000,BIN 00001000
2240 DATA BIN 00000000,BIN 00011
110,BIN 00000110,BIN 00001010,BI
N 00010010,BIN 00100010,BIN 0100
0000,BIN 10000000
2250 DATA BIN 00010000,BIN 00100
000,BIN 01000000,BIN 11111111,BI
N 01000000,BIN 00100000,BIN 0001
0000,BIN 00000000
2260 DATA BIN 00001000,BIN 00000
100,BIN 00000010,BIN 11111111,BI
N 00000010,BIN 00000100,BIN 0000
1000,BIN 00000000
2270 DATA BIN 00000001,BIN 00000
010,BIN 00000100,BIN 01001000,BI
N 01010000,BIN 01100000,BIN 0111
1000,BIN 00000000
2280 DATA BIN 00001000,BIN 00001
```

```
000,BIN 00001000,BIN 00001000,BI
N 01001001,BIN 00101010,BIN 0001
1100,BIN 00001000
2290 DATA BIN 10000000,BIN 01000
000,BIN 00100000,BIN 00010010,BI
N 00001010,BIN 00000110,BIN 0000
1111,BIN 00000000
```

Este módulo carga los ocho primeros caracteres definidos por el usuario con nueve flechas que se utilizarán para indicar la dirección de la presa.

Comprobación del módulo 6.2.1

Al ejecutar este módulo deberán rellenarse los espacios A a H en el área definida por el usuario, mediante flechas.

MODULO 6.2.2

```
1900>REM ***********************
1910 REM INSTRUCCIONES
1920 REM ***********************
1930 PRINT INK 3;AT 0,11;"INSTRU
CCIONES"
1940 PRINT "Este es un juego de
caceria."'"El terreno de caza es
 un"'"tablero de 18 x 30."''"La
presa es invisible."
1950 PRINT '"En cada turno la pr
esa hace un"'"movimiento secreto
. Este no"'"cambia durante una c
aceria."
1960 PRINT '"Puede ser de hasta
seis espacios"'"arriba o abajo y
 otros tantos a"'"la izquierda o
 a la derecha."
1970 PRINT '"Pulse una tecla par
a mas"'"instrucciones.": PAUSE 0
: CLS
1980 PRINT '''"Cada turno consis
te en:"
1990 PRINT '"1) Una invitacion a
 que entre su"'"estimacion de la
 posicion de la"'"presa."
2000 PRINT '"2) Aparecera una fl
echa en la"'"casilla especificad
a, indicando"'"la direccion de l
a presa."
2010 PRINT '"3) La presa se move
ra."
2020 PRINT '"Pulse una tecla par
a continuar.": PAUSE 0: CLS
2030 PRINT ''"Al inicio de cada
turno tiene la"'"oportunidad de
revisar como ha"'"ido la caceria
 hasta el momento."
2040 PRINT '"Esto se consigue en
trando 0"'"cuando se pide la coo
rdenada"'"VERT."
2050 PRINT '"Puede empezar la re
```

```
vision en"'"cualquier jugada pre
via pero"'"tiene un limite de re
vision de"'"20 jugadas por cacer
ia."
2060 PRINT '"Estas 20 revisiones
 pueden"'"tomarse de una vez o p
or partes."
2070 PRINT '"Pulsar una tecla pa
ra empezar": PAUSE 0: RETURN
```

Constituye la totalidad de las instrucciones del juego. El incluirlas o no en este juego, o en otros juegos que vaya a diseñar, depende de quién vaya a jugar.

MODULO 6.2.3

```
2080>REM ************************
2090 REM ESTABLECER DIFICULTAD
2100 REM ************************
2110 CLS
2120 PRINT '''"Hay un factor de
dificultad"'"previsto en el jueg
o."
2130 PRINT '"Consiste en un movi
miento"'"al azar de vez en cuand
o."
2140 PRINT '"El factor de dificu
ltad va de 0"'"hasta 10."
2150 PRINT '"0 es ningun movimie
nto al azar."
2160 INPUT INK 6;"Entre por favo
r el factor de"'"dificultad dese
ado:",E
2170 LET E=(11-E)*2+2+100*(E=0):
 CLS : RETURN
```

Este módulo establece el factor de dificultad. Si se entra un cero, el movimiento aleatorio se realiza tan sólo tras cien movimientos —y entonces el juego ya se habrá terminado.

MODULO 6.2.4

```
1000>GO SUB 2190: BORDER 0: INK
1: PAPER 7: PRINT AT 10,12;"CACE
RIA"
1010 INPUT "Quiere instruciones?
 (S/N)"'Q$: CLS : IF Q$="S" OR Q
$="s" THEN GO SUB 1910
1020 GO SUB 2080
1030 REM ************************
1040 REM VARIABLES
1050 REM ************************
1060 LET A=0: LET C=20: LET C1=0
: LET T=0: DIM M(100,4): DIM O$(
32)
1070 LET R1=6-INT (RND*13)
1080 LET R2=6-INT (RND*13)
1090 LET P1=INT (RND*18+1)
1100 LET P2=INT (RND*30+1)
```

Este módulo presenta la opción de obtener las instrucciones e inicializar las distintas tablas en las que se basará el programa.

MODULO 6.2.5

```
1110>REM ************************
1120 REM TABLERO INICIAL
1130 REM ************************
1140 CLS : LET A$="1234567890123
4567890123456789O": PRINT " ";A$
1150 FOR I=1 TO 18: PRINT A$(I):
 NEXT I: GO SUB 1370
```

Este módulo visualiza una serie de números alrededor del tablero, que sirven de guía al usuario.

MODULO 6.2.6

```
1370>REM ************************
1380 REM VISUALIZAR TABLERO Y
         JUGADA
1390 REM ************************
1400 PAPER 5: INK 1
1410 FOR I=1 TO 9: PRINT AT I*2-
1,1;" █ █ █ █ █ █ █ █ █ █ █ █ █
█ █";
1420 PRINT AT I*2,1;"█ █ █ █ █ █
 █ █ █ █ █ █ █ █ █": NEXT I
1430 INK 1: PAPER 7
1440 PRINT AT 19,0;O$;O$;O$
1450 IF T=0 THEN RETURN
1460 PRINT AT M1,M2;CHR$ (143+D)
1470 IF A<>1 THEN RETURN
1480 IF T/E=INT (T/E) THEN PRINT
 AT 21,0;"SE MOVIO AL AZAR!"
1490 RETURN
```

Este módulo visualiza el tablero sobre el que se realizará el juego. También visualiza la flecha definida por el usuario correspondiente, para un movimiento determinado e informa al usuario si se va a realizar un movimiento aleatorio.

Comprobación del módulo 6.2.6

Aunque muchas de las variables todavía no se han definido, si se coloca temporalmente un STOP en la línea 116∅ podrá ejecutar el programa, inspeccionar las instrucciones, especificar un nivel de dificultad y ver cómo se visualizan sobre la pantalla, el tablero y la retícula correspondiente.

```
1160 REM *************************
1170 REM ENTRADAS Y DIRECCIONES
1180 REM *************************
1190 INK 0: LET T=T+1: IF T>100
THEN CLS : PRINT AT 10,0;"LO SIE
NTO-SE ACABARON LAS JUGAD.": STO
P
1200 LET Q$=""
1210 PRINT AT 19,23;"JUG. ";T
1220 PRINT AT 21,17;"(0 P. REVIS
AR)"
1230 PRINT AT 19,1;"VERT:";: INP
UT M1: PRINT M1
1240 IF M1>18 OR M1<0 THEN PRINT
 ">FUERA MARGEN": PRINT AT 19,0;
Q$: GO TO 1210
1250: PRINT AT 20,0;Q$: IF M1=0
THEN GO SUB 1660: GO TO 1210
1260 PRINT AT 21,0;Q$;AT 20,1;"H
ORIZ.:";: INPUT M2: PRINT M2
1270 IF M2>30 OR M2<1 THEN PRINT
 ">FUERA MARGEN";AT 20,0;Q$: GO
TO 1260
1280 PRINT AT 21,0;Q$
1290 LET M(T,1)=M1: LET M(T,2)=M
2
1300 IF M1=P1 AND M2=P2 THEN GO
TO 1600
1310 LET M3=(M1<P1)-(M1>P1)+1
1320 LET M4=(M2<P2)-(M2>P2)+2
1330 LET D=M3*3+M4
1340 IF D>4 THEN LET D=D-1
1350 LET M(T,3)=D
1360 GO SUB 1370: GO SUB 1530: G
O TO 1190
```

Este módulo acepta la suposición del usuario para adivinar la posición de la presa y la guarda para una revisión posterior. Se calcula la flecha definida por el usuario correspondiente, para indicar la dirección de la posición real.

Comentario

Línea 129∅. El movimiento del jugador se guarda en la tabla M.

Líneas 131∅-135∅. Estas líneas convierten la dirección de la presa representada por P1 y P2, en la flecha correspondiente. Esto también se guarda en la tabla M en lugar correspondiente al movimiento actual.

Comprobación del módulo 6.2.7

Ahora ya puede definir una suposición de la posición de la presa y deberá aparecer una flecha indicando aproximadamente la dirección correcta —que puede comprobarse mediante P1 y P2—. El programa

no tiene que aceptar movimientos que caigan fuera de los límites del teclado.

MODULO 6.2.8

```
1500>REM ************************
1510 REM INCREMENTAR MOVIMIENTO
1520 REM ************************
1530 LET P1=P1+R1: LET P2=P2+R2
1540 IF T/E<>INT (T/E) THEN GO T
O 1570
1550 LET P1=INT (P1+RND*6+1): LE
T P2=INT (P1+RND*6+1)
1560 PRINT AT 21,0;"MOV. AL AZAR
"
1570 LET P1=P1+18*(P1<1)-18*(P1>
18)
1580 LET P2=P2+30*(P2<1)-30*(P2>
30)
1590 RETURN
```

Este módulo realiza los movimientos de la presa, incluyendo el movimiento aleatorio cuando sea necesario.

Comentario

Línea 1530. P1 y P2 son las coordenadas de la presa. R1 y R2 son los elementos del movimiento aleatorio que se suman a P1 y a P2 a cada movimiento, y que se establecen en la sección de variables.

Línea 1540. La variable E se coloca de acuerdo con el nivel de dificultad. Cada E movimientos se realiza el movimiento aleatorio, aumentando la dificultad.

Líneas 1570-1580. Estas dos líneas aseguran que si la presa se mueve fuera del tablero en cualquier dirección, vuelve a aparecer por el otro lado del mismo —un efecto envolvente.

Comprobación del módulo 6.2.8

Ahora ya podrá entrar una serie de movimientos.

MODULO 6.2.9

```
1600>REM ************************
1610 REM EXITO POR FIN
1620 REM ************************
1630 PRINT AT 10,13;"CAZADO!"
1640 INPUT "Otro juego? (S/N)",Q
$: IF Q$="S" THEN GO TO 1030
1650 STOP
```

Este módulo anuncia el éxito del jugador cuando se encuentra la presa.

Comprobación del modulo 6.2.9

En lugar de esperar a descubrir la presa, ejecute el juego y, cuando se le pida que entre la posición, detenga el programa con un STOP y visualice P1 y P2 en modo directo. Inicialice de nuevo el programa con un GOTO 119Ø y entre las coordenadas correctas, con lo que se producirá la llamada de este módulo.

MODULO 6.2.10

```
1660>REM ***********************
1670 REM REVISION DE JUEGO
1680 REM ***********************
1690 LET A=1: PRINT AT 19,0;O$;O
$;O$
1700 IF C1>=C THEN GO TO 1780
1710 PRINT AT 6,13;"REVISION"
1720 PRINT AT 8,4;"REV. PERMITID
A ";C;" JUGADAS"
1730 PRINT AT 10,8;"UTILIZADAS "
;C1
1740 PRINT AT 19,1;"ULTIMA JUG.
FUE LA:";T-1
1750 PRINT AT 21,1;"ENTRAR PRIM.
 JUG. A REVIS.:";: INPUT T1: PRI
NT T1
1760 FOR J=T1 TO T-1
1770 LET C1=C1+1
1780 IF C1>C THEN PRINT AT 10,4;
"AGOTADAS LAS REVISIONES": PAUS
E 200: GO TO 1860
1790 LET M1=M(J,1): LET M2=M(J,2
): LET D=M(J,3)
1800 GO SUB 1370
1810 PRINT AT 19,6;"REVISION JUG
ADA:";J
1820 PRINT AT 21,20;"(0 SALIR)"
1830 INPUT Q$: PRINT AT 19,0;O$;
O$;O$
1840 IF Q$="0" THEN GO TO 1860
1850 NEXT J
1860 LET T=T-1
1870 LET M1=M(T,1): LET M2=M(T,2
): LET D=M(T,3)
1880 GO SUB 1370
1890 LET A=0: RETURN
```

Quizá se haya preguntado por qué cada movimiento y su flecha correspondiente se guardan en la tabla M. Este módulo utiliza los datos guardados en M para reconstruir los movimientos y las claves dadas anteriormente para su reexamen.

Comprobación del módulo 6.2.10

Ahora ya podrá llamar al módulo que realiza la repetición de los movimientos previos, entrando un cero cuando se le pida la coordenada vertical. Si este módulo es correcto, el programa ya está listo para su utilización.

Resumen

Este juego es un clásico ejemplo de un programa que se va «vistiendo». El juego empezó con algo muy pequeño que cabía en un K de memoria del ZX81. Era divertido de jugar por lo que el tablero se fue ampliando para que fuese más interesante. Entonces, debido a que llegó a ser demasiado difícil, se añadió la función de repetición. Finalmente, el movimiento aleatorio completó el proceso. La moraleja es que siempre vale la pena jugar con ideas sencillas, incluso aunque parezcan demasiado triviales, ya que con un poco de guarnición podría tener el sucesor de Invasores del Espacio en sus manos.

Posibles mejoras

1) Una mejora definitiva sería la posibilidad de un final del juego que permitiese volver a ver los movimientos, junto con la posición de la presa a cada turno.

6.3 Clasificación de palabras

Si le gustan los juegos de palabras, y no le importa esperar un par de horas, necesitará una rutina de clasificación de textos. Es un eficiente método de clasificación capaz de disminuir el tiempo necesario para ordenar elementos.

MODULO 6.3.1

```
1460>REM ************************
1470 REM CLASIFICACION
1480 REM ************************
1500 LET A=INT (LN N/LN 2): LET
F=2↑A-1
1510 LET F=INT (F/2): IF F=0 THE
N RETURN
1520 LET D=N-F: LET B=1
1530 LET A=B
```

```
1540 LET E=A+F: IF A$(A)<A$(E) T
HEN GO TO 1570
1550 LET B=B+1: IF B>D THEN GO T
O 1510
1560 GO TO 1530
1570 LET T$=A$(A): LET A$(A)=A$(
E): LET A$(E)=T$
1580 LET A=A-F: IF A<1 THEN GO T
O 1550
1590 GO TO 1540
```

Este módulo merece un estudio detallado. Hasta ahora hemos utilizado una variedad de métodos para evitar la clasificación del material, prefiriendo encontrar el lugar correcto para un elemento cada vez. Este lugar se guarda en una tabla de punteros en lugar de poner por orden la tabla principal. Cuando deben entrarse un gran número de elementos pequeños esto puede llegar a ser un proceso engorroso, con una tediosa espera después de cada entrada, hasta que se encuentre el lugar correcto. En estas circustancias es mucho mejor entrar los datos en cualquier orden en que aparezcan y clasificar después los elementos que se han entrado.

Muchos programas utilizan una clasificación y muchos de estos programas de clasificación utilizados en casa llevan programas realmente ineficientes. No se trata meramente de un problema de estilo; una clasificación ineficiente en un programa bastante sencillo puede ser fácilmente el proceso más largo, requiriendo un pesado período de espera cada vez que se entran los nuevos elementos de los datos. Un programa destinado al predecesor del Spectrum, el ZX81, publicado en una importante revista, necesitó 2Ø minutos para clasificar cien elementos de datos por orden alfabético. El método utilizado en este módulo reduce este tiempo a tres minutos.

El método de clasificación utilizado se conoce con el nombre de Shell-Metzner. Funciona a base de recorrer los datos que hay que clasificar, comparando parejas que deberán intercambiarse si están en orden incorrecto. Para empezar, la distancia entre los elementos de cada pareja es la mayor potencia de dos que es menor que el número total de datos. La clasificación empieza con el primero y compara el dato en esta posición, con, por ejemplo el que está en la posición 64, en el caso de 1ØØ datos. Los datos se van recorriendo hasta que no hay más posibles parejas que estén a esta distancia. El intervalo entonces se reduce y la comparación de parejas empieza de nuevo por el principio.

Una de las mejores formas para comprender este método de clasificación con detalle, es escribir los números del 1 al 2Ø sobre tiras de papel y después, con una libreta a mano para registrar las variables, realizar la clasificación, basándose en este módulo como si se tratase del Spectrum. Cuando haya terminado quizá siga encontrando

difícil ver el porqué funciona, pero así es y es uno de los métodos de clasificación disponibles más eficientes.

Comentario

Líneas 15ØØ-151Ø. Estas dos líneas encuentran la mayor potencia de 2 que es menor que el número total de elementos que hay que clasificar —de hecho menos uno.

Línea 152Ø. D es un indicador; si la separación actual entre los elementos de las parejas es F, entonces el primer elemento del último par posible estará en N-F. B guarda el primer elemento de la pareja actual.

Línea 154Ø. A$ se dimensiona con N elementos. Obsérvese que los elementos se están clasificando en orden inverso de su longitud, en el caso de este módulo.

Línea 157Ø. Esta línea realiza el intercambio si la línea 154Ø ha descubierto que los elementos de la pareja están en orden incorrecto.

Línea 158Ø. En el caso de que un elemento haya sido intercambiado hacia atrás, por ejemplo el elemento 82 por el 5Ø, el módulo vuelve hacia atrás para examinar la pareja a la cual corresponde ahora el segundo elemento es decir 18/5Ø.

Comprobación del módulo 6.3.1

No podrá comprobar esta rutina hasta que desarrolle un programa que necesite utilizar una clasificación. El programa puede cambiarse fácilmente para que clasifique números en lugar de textos, dimensionando una variable A con N elementos.

Conclusión

Si ha analizado la totalidad de los programas de este libro, aparte de merecer una medalla de algún tipo, ahora será el orgulloso poseedor de una biblioteca de programas. Ciertamente, no se trata de la mayor biblioteca de la historia de los ordenadores, pero contiene las herramientas para emprender una amplia variedad de tareas con sólo adaptar los programas a sus propias exigencias específicas. Además de esto, es una indicación, una pista —no más— de lo que el Spectrum puede llegar a realizar para usted.

Los programas de este libro son ideas que funcionaron para mí, que me interesaron, que resolvieron los problemas que tuve. Ya que son mías, a algunas de ellas quizá les falte algo cuando las aplique a sus propios problemas. En este caso cámbielas, mutílelas, descompóngalas en piezas. Serán mucho mejores cuando las haya hecho a su medida.

Indice de módulos

Los números que hay a continuación de las entradas del índice se corresponden con los números de los módulos.

Programas prácticos para el Spectrum está basado en una colección de sólidos y sofisticados programas que tratan sobre temas como almacenaje de datos, finanzas, cálculos, gráficas, gestión doméstica y educación.

Cada uno de los programas está explicado con detalle, línea a línea, a la vez que construido a base de subrutinas y módulos de propósito general que, una vez comprendidos, pueden formar la base de cualquier otro programa que necesite escribir.

Con el análisis explicativo de cada subrutina aparecen técnicas avanzadas de programación. El resultado no sólo es el avance en cuanto a sus técnicas de programación, sino también el obtener una gran variedad de programas prácticos de aplicación que de otra forma tan sólo serían accesibles para aquellos que estuvieran dispuestos a comprar cassettes o los que fueran capaces de escribir programas sustanciales por sí mismos.

El autor, Davis Lawrence, es un colaborador habitual de *Popular Computing Weekly.*

Editorial Gustavo Gili, S. A.
Rosellón, 87-89
08029 Barcelona